Lumenera Network Camera
API Reference Manual
Release 1.8.1.16

Lumenera corporation

License Agreement (Software):

This Agreement states the terms and conditions upon which Lumenera Corporation ("Lumenera") offers to license to you (the "Licensee") the software together with all related documentation and accompanying items including, but not limited to, the executable programs, drivers, libraries, and data files associated with such programs (collectively, the "Software").

The Software is licensed, not sold, to you for use only under the terms of this Agreement.

Lumenera grants to you the right to use all or a portion of this Software provided that the Software is used only in conjunction with Lumenera's family of products.

In using the Software you agree not to:

a) decompile, disassemble, reverse engineer, or otherwise attempt to derive the source code for any Product (except to the extent applicable laws specifically prohibit such restriction);

b) remove or obscure any trademark or copyright notices.

Limited Warranty (Hardware and Software):

ANY USE OF THE SOFTWARE OR HARDWARE IS AT YOUR OWN RISK. THE SOFTWARE IS PROVIDED FOR USE ONLY WITH LUMENERA'S HARDWARE AND OTHER RELATED SOFTWARE. THE SOFTWARE IS PROVIDED FOR USE "AS IS" WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED BY LAW, LUMENERA DISCLAIMS ALL WARRANTIES OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, QUALITY AND FITNESS FOR A PARTICULAR PURPOSE. LUMENERA IS NOT OBLIGATED TO PROVIDE ANY UPDATES OR UPGRADES TO THE SOFTWARE OR ANY RELATED HARDWARE.

Limited Liability (Hardware and Software):

In no event shall Lumenera or its Licensors be liable for any damages whatsoever (including, without limitation, incidental, direct, indirect, special or consequential damages, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use or inability to use this Software or related Hardware, including, but not limited to, any of Lumenera's family of products.

Product Warranty

Lumenera Corporation warrants to the original purchaser that our cameras are guaranteed to be free from manufacturing defects for a period of one (1) year from the original date of purchase.

Should the unit fail during the warranty period, Lumenera will, at its option, repair or replace the failed unit. Repaired or replaced units will be covered under warranty for the remainder of the original one (1) year warranty period.

Copyright © 2009

This warranty does not apply to units that, after being inspected by Lumenera, have been found to have failed due to customer abuse, accidents, mishandling, tampering/alteration, improper installation, improper power source, negligence, opening of the enclosure, or if the serial number has been removed or damaged. This warranty does not cover labor or incurred charges required in removing or installing the unit, any business interruption, loss of profits/revenues, or any consequential damages.

Units returned to Lumenera beyond the warranty period will be repaired, if possible, and all appropriate material and labor charges will apply. The repaired part will be covered under an additional 90-day warranty.

Any returning product, specifically those being returned under warranty, must follow the Returned Material Authorization (RMA) process. Units must be properly packaged, in original packing when possible. Lumenera will not cover damage sustained in shipping due to improper packing.

All shipping charges incurred for the return of failed units are the customer's responsibility, including shipping, broker fees, duties and taxes. Once the unit is repaired or replaced, Lumenera will pay the shipping charges to return the unit back to the customer.

For RMA instructions, please refer to our website at www.lumenera.com.

RoHS/WEEE Compliance Statement

The Restriction of Hazardous Substances in Electrical and Electronic Equipment (RoHS) Directive was passed into law by the European Union (E.U.). It affects manufacturers, sellers, distributors and recyclers of electrical and electronic equipment containing lead, cadmium, mercury, hexavalent chrome, polybrominated biphenyl (PBB) and polybrominated diphenyl ether (PBDE). After July 1, 2006 the use of these materials will be banned in new products sold in Europe. The RoHS Directive complements the WEEE Directive. China is expected to adopt similar legislation within a similar timeline.

The Waste Electrical and Electronic Equipment Directive (WEEE) aims to reduce the waste arising from electrical and electronic equipment and to improve the environmental performance of all those involved in the life cycle of these products.

Lumenera is committed to protecting people and the environment and we are working on identifying any materials used in our processes that could pose a potential hazard to our employees, customers or the environment.

For this reason we are committed to have all our products comply with the RoHS and WEEE directives. We are constantly improving our compliance with these directives. For more information on our compliance or to track our progress please refer to our website.

Copyright © 2009

Table of Contents

INTRODUCTION 1
1.1 THE LUMENERA NETWORK CAMERA FAMILY 1
1.2 WHERE TO FIND DOCUMENTATION 2
1.3 TECHNICAL ASSISTANCE 2
NETWORK CAMERA API 3
2.1 THE LUMENERA NETWORK CAMERA API 3
2.2 CAMERA FEATURES AND SETTINGS 4
2.3 SUPPORTED NETWORK PROTOCOLS 6
2.4 PROGRAMMING INTERFACES 7
2.4.1 XML-RPC Environment 7
2.4.2 CGI/HTTP/Javascript Environment 9
2.4.2.1 No Process Header (NPH) commands 11
2.4.2.2 Response on Success 12
2.4.2.3 Response on Failure 12
2.4.2.4 Access Levels 14
2.4.3 Telnet Environment 15
API FUNCTION REFERENCE 16
3.1 SUMMARY OF API FUNCTIONS 16
3.2 API FUNCTION DESCRIPTION 17
add_parameter 17
ftp_images 19
ftp_stream_last_error 22
ftp_stream_status 23
ftp_stream_stop 24
ftp_stream_stop_all 25
get 26
get_settings 27
image 28
load_settings 30
ptz_command 31
ptz_preset 32
put_settings 33
put_settings_from_property 34

reset 35
save_settings 36
set 37
start 38
stop 39
strobe 40
test_email_alarm 41
udp_stream 42
udp_stream_stop 44
udp_stream_stop_all 45
version 46
video 47
API PROPERTY REFERENCE 49
4.1 API PROPERTY DESCRIPTION 49
admin_password 49
alarm_date_last 49
alarm_delay 50
alarm_enable_email 50
alarm_enable_ftp 51
alarm_enable_udp 51
alarm_ftp_ramerate 52
alarm_ftp_passive 52
alarm_ftp_password 52
alarm_ftp_path 52
alarm_ftp_post_trigger 53
alarm_ftp_pre_trigger 53
alarm_ftp_server 53
alarm_ftp_user 53
alarm_udp_client 54
alarm_udp_port 54
analytics_alarm_enable 54
analytics_email_script 55
analytics_enable 55
analytics_view_select_model 55
aperture 56
aperture_hw_max 57
aperture_hw_min 57

Copyright © 2009

aperture_max 57
aperture_min 58
auto_algorithm 58
auto_height 59
auto_jpeg_quality 59
auto_width 60
auto_window_enable 60
auto_x 61
auto_y 61
autoexposure 61
autoexposure_knee 62
autoexposure_target 63
autogain 63
autogain_knee 63
autoiris 64
boot_version 64
brightness 65
color_conversion 65
color_preset 65
contrast 65
crosshair_enable 66
current_broadcast_address 66
current_fqdn 66
current_gateway 67
current_ip 67
current_nameserver 67
current_netmask 67
date 68
date_utc 68
day_mode 68
daynight_bw_mode 69
daynight_gain_boost 69
daynight_gain_in_threshold 69
daynight_gain_limit 70
daynight_gain_out_threshold 70
daynight_mode 71
default_image_params 71

description 71
dhcp_fallback 72
dhcp_timeout 72
dns_domain_name 72
edge_enhancement 73
ethaddr 73
exposure 73
exposure_bias 74
focus 74
focus_max 74
focus_min 74
frame_buffer_enable 75
framerate 75
gain 75
gain_blue 76
gain_blue_max 76
gain_blue_min 77
gain_green 77
gain_green_max 77
gain_green_min 78
gain_red 78
gain_red_max 78
gain_red_min 79
gamma 79
gateway 79
h264_enable 80
height 80
image_stabilizer 80
imager_mono 81
intensity 81
io_in_alarm_enable 81
io_in_debounce_time 82
io_in_trigger_edge 82
io_in_trigger_neg 82
ipaddr 82
iris_type 83
ir_shuttle_position 83

Copyright © 2009

is_running 83
jpeg_buffer_size 83
jpeg_quality 84
led 84
lighting_frequency 84
link_local_ip 85
low_light_criteria 85
low_light_settings 86
low_light_stable_duration 86
luminance 86
MAC 87
max_height 87
max_jpeg_size 87
max_subsample 88
max_width 88
maximum_exposure 88
maximum_framerate 88
maximum_gain 89
maximum_gain_limit 89
median_filter 89
mem_free 90
min_jpeg_size 90
minimum_exposure 90
motion_alarm_area 90
motion_alarm_enable 91
motion_email_recipient 91
motion_email_script 92
motion_email_server 92
motion_email_subject 93
motion_gap 93
motion_map 93
motion_map_height 94
motion_map_width 94
motion_map_windowed 95
motion_sensitivity 95
motion_skip 96
motion_x 96

Copyright © 2009

motion_y 97
name 97
nameserver 97
netmask 98
normal_light_criteria 98
normal_light_settings 98
normal_light_stable_duration 99
ntp_servers 99
number_of_buffers 99
number_of_buffers_raw 100
output_select 100
privacy_map 101
privacy_map_enable 102
privacy_map_height 102
privacy_map_width 102
projected_network_bandwidth 103
projected_network_framerate 103
ptz_enable 104
ptz_protocol 104
ram_size 105
raw_mode 105
rtp_enable 105
rtp_framerate 106
rtp_packet_size 106
rtp_port 106
saturation 107
saturation_bias 107
serial_echo 107
serial_flow 108
serial_speed 108
serial_use 108
show_motion_detected 108
show_motion_windows 109
strobe_length 109
subsampled 110
target_jpeg_size 110
telnetd 110

Copyright © 2009

temperature *111*
timezone *111*
use_dhcp *111*
use_link_local *112*
user_password *112*
video_mode *113*
watchdog_codec *113*
watchdog_enable *114*
watchdog_http *114*
watchdog_ping *114*
whitebalance *115*
whitebalance_preset *115*
width *116*
x *116*
x_mirror *117*
y *117*
y_mirror *117*
APPENDIX: INTERPRETING EXIF HEADERS 118
A1.1 GENERAL INFORMATION ON EXIF HEADER FORMAT 118
A1.2 DETAILED EXAMPLE OF EXIF HEADER 118
GLOSSARY 120

List of Tables

Table 2-1 URI Escape Code for Special Characters 10
Table 2-2 Available Access Levels in Network Cameras 14
Table 3-1 Summary of Camera API Functions 16
Table 4-1 `MakerNote` Field Descriptions in the EXIF header 119

Copyright © 2009

1

Introduction

1.1 The Lumenera Network Camera Family

The Lumenera Network camera Application Programming Interface (API) provides tools for users to access and customize advanced functionality governing a full suite of camera control and image processing features. A Function and Property (API) Reference document is stored on every camera in HTML format for viewing using a web browser. This manual provides additional information on these functions and properties.

Lumenera Network Cameras provide high resolution images and are designed to satisfy both general-purpose and high-end surveillance applications while delivering outstanding image quality. The cameras use on-board JPEG compression to deliver high quality images over a standard 10/100baseT Fast Ethernet connection. Most models support Power over Ethernet (PoE). In video surveillance applications, Lumenera cameras work with an extensive list of third-party NVR/DVR software and hardware to provide complete video management solutions. Support for different camera models is kept uniform using a common Application Programming Interface to the greatest extent possible.

Lumenera's product line is unique in its breadth, with resolutions ranging from VGA to 11 megapixels along with a choice of several CMOS and CCD image sensors in both color and monochrome types. Most models use C/CS- mount lenses commonly found in CCTV and factory automation, and are available with an integrated day/night filter option for use with active-infrared illumination. For demanding applications such as low-light surveillance, traffic enforcement, and industrial process monitoring, Lumenera provides high-sensitivity, low-noise CCD's with global shutter. Large-format models use off-the-shelf 35-mm motorized lenses to provide exceptional image quality and sensitivity.

The Li-series intelligent cameras provide additional features including embedded rules-based video content analysis, audio input and output, and analog video output for use during camera installations.

The description of the Application Programming Interface (API) applies specifically to firmware version 1.8.1.16 Extensions specific to the Li series intelligent cameras refer to firmware version 2.3.1.2.

Copyright © 2009

1.2 Where To Find Documentation

All camera documentation can be found in the <Install Directory>\Documentation folder. The documentation that is provided includes this manual, the Network Camera User's manual, camera Application Notes and White Papers.

To obtain the latest software release, Application Notes, answers to Frequently Asked Questions (FAQs), Knowledge Base access and other technical information please visit our website at:

www.lumenera.com

Note: A user name and password is required to access the Knowledge Base. If you do not yet have one, or require a new one, please send an email to support@lumenera.com to receive a new password.

1.3 Technical Assistance

If you need assistance with the installation or use of the software, or, if you need help with general camera operation, please contact the Technical Assistance Centre (TAC) via email at:

support@lumenera.com

or by phone at +1-613-736-4077 (press 1 from the auto attendant).

To obtain the latest software release and other technical information you may visit our technical support website at:

http://www.lumenera.com/support/index.php

Our support website contains technical information available to the general public such as Frequently Asked Questions (FAQ's). For our Lumenera customers we provide a Knowledge Base with more product specific solutions and a Download Centre for customers to obtain the most recent software releases.

As a customer, you will need to provide the TAC with some basic information to gain access to the customer Knowledge Base and the Download Centre. Please provide the following details via email to support@lumenera.com to obtain a user name and password:

- Your name, Company Name, address and telephone number.
- Your camera model and serial number.
- Your purchase information (e.g. did you purchase from an OEM or distributor?)
- Upon providing the above information, you will receive your access information via email from a TAC representative.

Copyright © 2009

2

Network Camera API

2.1 The Lumenera Network Camera API

The API supports several functions that initiate tasks such as the transmission of images or saving settings on the camera. In addition, the API includes properties governing all camera settings (e.g. autoexposure, autogain) that can be altered using the `set` function and queried using the `get` function.

Entering the IP address of any accessible camera in a web browser allows the user to adjust most of the camera functions embodied in this API using a graphical user interface (GUI). The complete set of API commands and settings can be issued by entering an appropriate URL in a web browser or through various console interfaces such as a Telnet session or a serial connection the camera's RS232 interface.

For custom applications, the API commands can be applied programmatically. Access the camera by using either the Visual Basic or C++ XML-RPC libraries provided on the Network Camera's installation CD-ROM, by using either Java or Python scripts or by using the HttpWebRequest class in C#.NET. Simple examples using many of these languages are provided on the CD-ROM provided as part of the Lumenera Network camera SDK.

The API includes an optional administrator's (admin) password that grants full access to all functions and settings available on the Network camera. In addition, one may set a user password that grants access only to video, single images, the firmware version information or simple queries using the `get` function.

Some of the more advanced camera features that are available using the API include the ability to configure the JPEG quality settings and the on-board image buffer. Using the `video` command, it is possible to request pre-event images from the on-camera memory. The camera can also transfer pre- and post-event images to an FTP server for a user-selected duration. The image file transfer using FTP can be initiated by a client. Alternatively, the FTP session can begin automatically upon an alarm. This alarm can be triggered by the camera's built-in video motion detection or by a signal received on the GPIO input port of the camera.

Copyright © 2009

The Network Camera API provides a full set of functions and properties that allows one to completely control and configure the camera based on your application needs.

2.2 Camera Features And Settings

The Lumenera Network cameras provide many features that can be applied to any application environment to achieve the best image with the best data quality available. A list of features supported by Lumenera Network Cameras follows.

- Auto Brightness Control (ABC)
- Autogain or fixed gain control
- Autoexposure or fixed exposure control
- 60 Hz / 50 Hz lighting synchronization
- Configurable ABC iris, gain, & exposure knee points
- Auto iris enable or disable
- 4-wire DC (direct drive) auto iris on CS-mount models.
- Configurable brightness targets
- Four histogram-based ABC algorithms
    - Average
    - Median (similar to Average, but ignores extreme bright and dark)
    - Peak detect (regulate only on brightest areas)
    - Dark detect (regulate only on darkest areas)
- White balance
    - Continuous auto white balance
    - Single shot white balance
    - User-defined AutoWindow selection
- Defines region of interest for white balance and Auto Brightness Control (ABC)
- EF-mount (Canon) SLR lens control:
    - For large-format models Le259C & Le11059C
    - Remote control of focus
    - Auto iris: user-specified day time and night time lens aperture.
- Image processing

Copyright © 2009

  - Saturation (monochrome through to bright color)
  - Contrast
  - Sharpening
  - Brightness
- Light source color temperature corrections with color gain limits
  - Default
  - Daylight
  - Fluorescent
  - Halogen
  - Wide
- Gamma (mid-tone brightness)
- JPEG quality
- Frame rate transmitted over the network
- Number of buffers (images) in on-camera memory
- Geometry (Region of Interest)
  - Configurable width, height, and location
  - Subsampling (decimation) and/or binning
  - Horizontal (x-mirror) or vertical (y-flip) image flip
- Privacy map – blocks private areas from view
- Day/Night functions
  - With '-DN' option, infrared-cut filter switches to infrared-transparent glass in low light
  - Define day/night thresholds based on exposure, gain, and/or luminance properties.
- Alarms
  - Video motion detection in specific checkerboard areas, a.k.a. the motion map, can generate an alarm
  - Alarm input signal can be detected on the optically isolated dry-contact GPIO input
  - Email notification on alarm
  - FTP image transfer to a specified server on alarm, including pre-alarm images.
  - UDP alarm notification to a specified client

Copyright © 2009

    - User can edit Linux script to perform other functions upon alarms
- Static or dynamic IP settings
- Date/time/time zone settings
    - Can specify one or more NTP (Network Time Protocol) servers
- Passwords
    - Admin (full rights)
    - User (view images and simple queries)
- Enable/disable telnet daemon
- Enable/disable serial console (RS232 port)
- Watchdogs – periodic built-in self tests
- Firmware upgrade through web browser interface
- Reset default image settings
- Save user settings to camera memory
- Reboot camera

Many of these functions are accessible through the camera's web interface. All of them are accessible through the camera's API interface. Subsequent sections of this manual explain how to access and control these features through the API. The Lumenera Network Camera User's manual explains how these features can be accessed through the camera's web interface.

2.3 Supported Network Protocols

Communication to the Network camera can be done using several standard Ethernet/Network based protocols. Some of the standard protocols supported by the camera include:

TCP/IP: Transmission Control Protocol/Internet Protocol suite, used for command and control of the camera programmatically

FTP (client only): File Transfer Protocol, used for transferring image data from the camera to an FTP server using either a FTP client application or programmatically

XML-RPC: eXtensible Markup Language – Remote Procedure Call, used for command and control of the camera programmatically

HTTP: HyperText Transfer Protocol, used for transferring image data from the camera using either a web browser, media player, or a custom application programmatically

Copyright © 2009

CGI: Common Gateway Interface, used for command and control of the camera using external application software such as web browsers or media players.

Telnet: TELecommunication NETwork protocol, used for command and control of the camera

UDP: User Datagram Protocol, used for streaming image data with higher throughput. The trade-off compared to TCP/IP is that receipt of all packets is not guaranteed.

ARP: Address Resolution Protocol, used for finding and resolving network addresses

DHCP: Dynamic Host Control Protocol, used to get camera's IP, gateway, subnet mask and DNS servers.

DNS: Domain Name System, used to translate domain names, email servers and other information into addresses

SMTP (client only): Simple Mail Transfer Protocol, used to transmit emails from the camera (used for alarm notification)

ICMP: Internet Control Message Protocol, used for ping requests and error messages

Zeroconf: ZERO CONFiguration networking, used to setup a quick, simple network without the need for additional network servers

2.4 Programming Interfaces

Accessing the Network camera's API functions and properties is accomplished through any of the following interfaces:

XML-RPC (see Section 2.4.1 for more details)

CGI/HTTP/Javascript (see Section 2.4.2 for more details)

Telnet (see Section 2.4.3 for more details)

The following sections will describe how to access the camera's functions and properties within their respective interface.

2.4.1 XML-RPC Environment

XML-RPC is a form of remote procedure calls that can be done over the Internet using XML as an encoding format for the requests. An XML-RPC message request is an HTTP-POST request where the body of the request is an XML formatted message. More information about XML-RPC can be found at www.xmlrpc.com.

All Network camera functions can be accessed through the XML-RPC library. All camera API functions are prefixed with "le." when accessed through this

interface. They communicate through port 80 using the camera's RPC interface URI (Uniform Resource Identifier).

The Network Camera SDK installation CD-ROM includes a XML-RPC library, called XML-RPC++, that can be used in a Microsoft Windows based application.

An example of an XML-RPC request using this library is shown below. This example queries the camera for the type of imager used.

```
BOOL GetImagerType(CStringA *csImager)
{
  XmlRpcValue args;
  XmlRpcValue result;
  CStringA csCommand;
  CString csDebug;
  BOOL bReturn = TRUE;
  CHAR *pcBuffer;
  XmlRpcClient *xmlClient;

  xmlClient = new XmlRpcClient("192.168.2.24", 80, "/cgi-
bin/rpc");
  if (xmlClient != NULL)
  {
    // Need to add 1 for '\0' at the end of the string
    pcBuffer = new CHAR [20];

    // setup function arguments
    strcpy_s(pcBuffer, 20, "imager_type");
    args[0] = XmlRpcValue(pcBuffer);

    // setup function command
    pcBuffer = new CHAR [10];
    strcpy_s(pcBuffer, 10, "le.gets");
    if(!xmlClient->execute(pcBuffer, args, result))
    {
      bReturn = FALSE;
    }
    // copy return information into function arguments
provided
    *csImager = static_cast<std::string>(result).c_str();

    // Clean up and destroy object
    xmlClient->close();
    delete xmlClient;
  }
```


Copyright © 2009

```
  return(bReturn);
}
```

Note: Image capturing functions should use CGI/HTTP commanding instead of XML-RPC as the latter interface results in a perceived decrease in camera performance. Also note, when using the provided XML-RPC++ library, it is recommended to create one object per XML-RPC request and to delete the object once complete to improve the stability of the library within your application.

2.4.2 CGI/HTTP/Javascript Environment

CGI, HTTP and Javascript environments all use the same protocol for accessing the Network camera API. The protocol used is HTTP on port 80 using the GET method, the same method that a web browser uses to access a web page. For a description of the CGI standard see http://hoohoo.ncsa.uiuc.edu/cgi/.

Note: Instead of a GET request, an HTTP POST request may be used. If so, the mime-type must be multipart/form-data but a GET request is the recommended and supported request format. Examples follow.

```
     GET /cgi-bin/nph-video HTTP/1.0
     GET /cgi-bin/version HTTP/1.0
     GET /cgi-bin/get?exposure HTTP/1.0
     GET /cgi-bin/set?x=10&y=15&width=150&height=100
HTTP/1.0
```

CGI commands are executed by constructing and requesting a specific Uniform Resource Indicator (URI) with a directory name of /cgi-bin/. The directory name is followed by a base name that specifies the command to be executed.

Arguments are appended to the base name to complete the URI. There is no actual message body. This means that the URI can be bookmarked in a web browser. The camera returns a web page presenting the results of the request.

Arguments are specified as they would be in any standard HTML form submission. A command is separated from its arguments with a question mark (`?`) and arguments are separated with an ampersand (`&`). Each argument is of the form `argument=value`.

The content of strings that include special characters must use conventional URL or URI escape codes. In particular, the characters in Table 2-1 may require substitution with the URL and URI escape codes indicated.

Table 2-1 URI Escape Code for Special Characters

Character	URI Escape Code	Character	URI Escape Code
SPACE	%20	?	%3F
$	%24	@	%40
%	%25	[	%5B
&	%26	\	%5C
/	%2F	]	%5D
:	%3A	^	%5E
;	%3B	`	%60
<	%3C	{	%7B
=	%3D	|	%7C
>	%3E	}	%7D
		~	%7E

Below are some examples of how to access the camera's API interface. Where indicated, replace "`CamIP`" with the camera IP address (e.g. `192.168.2.89`) or the DNS name of the camera.

Examples on how to access functions through the web interface:

Launch MJPEG video stream:

```
http://CamIP/cgi-bin/nph-video
```

or,

```
http://CamIP/cgi-bin/nph-video?type=multipart/x-mixed-replace
```

Query firmware & hardware version:

```
http://CamIP/cgi-bin/version
```

Get camera settings information:

```
http://CamIP/cgi-bin/get_settings
```

Save settings to persistent memory:

```
http://CamIP/cgi-bin/save_settings
```

Restore settings from memory:

```
http://CamIP/cgi-bin/load_settings
```

Reboot the camera:

Copyright © 2009

```
http://CamIP/cgi-bin/reset
```

All camera properties use the `get` and `set` camera API functions. Below are some examples of how these properties can be accessed:

Set fixed gain and exposure:

```
http://CamIP/cgi-bin/set?gain=1&exposure=10
```

Enable auto-brightness:

```
http://CamIP/cgi-bin/set?autogain=1&autoexposure=1
```

Enable automatic white balance:

```
http://CamIP/cgi-bin/set?whitebalance=continuous
```

Get the camera gain:

```
http://CamIP/cgi-bin/get?gain
```

Get the camera MAC address:

```
http://CamIP/cgi-bin/get?MAC
```

Set custom day/night modes:

Parameters to set:

```
low_light_settings=ir_shuttle_position=1;auto_algorithm=pea
k;lighting_frequency=0;maximum_exposure=4;autoexposure_targ
et=20;autoexposure=1;autogain=1;
```

Command with URL encoded special characters:

```
http://CamIP/cgi-
bin/set?low_light_settings=ir%5Fshuttle%5Fposition%3D1%3Bau
to%5Falgorithm%3Dpeak%3Blighting%5Ffrequency%3D0%3Bmaximum%
5Fexposure%3D4%3Bautoexposure%5Ftarget%3D20%3Bautoexposure%
3D1%3Bautogain%3D1%3B
```

2.4.2.1 No Process Header (NPH) commands

You may also prefix any CGI command name that is specific to the camera API (video, image, get, set, get_settings, version, etc.) with “nph-”. A NPH CGI program starts sending data immediately, rather than buffering it to construct the Content-Length HTTP header. By using these commands, the requests bypass the web server by writing directly to the HTTP socket. The NPH command thus has a significantly lower latency. The NPH command format is recommended for all image and video transfers. Examples follow.

Single JPEG image:

```
http://CamIP/cgi-bin/nph-image.jpg
```

MJPEG image stream:

```
http://CamIP/cgi-bin/nph-video
```

2.4.2.2 Response on Success

If the API function call succeeds then `templates/success.tmpl` is processed and returns a `Success` message formatted as HTML. Functions that return binary results, such as `/cgi-bin/nph-image.jpg`, do not use the response template mechanism unless there is an error. The format of the success template places the return value in a `<!--CAMERA -->` tag. This is followed by an HTML formatted body that includes a `<function> executed successfully` message, the return value (if any), and a hyperlink to the index page of the camera.

Example: query firmware version

```
http://CamIP/cgi-bin/version
```

Return value:

```
Lumenera/explorer2 r5281 hw1.1 1.8.0.10 : le075c-dn -- Fri,
21 Sep 2007 13:31:21 -0400
```

Success template response:

```
<html>
<head>
<title>Success</title>
<!-- CAMERA Lumenera/explorer2 r5281 hw1.1 1.8.0.10 :
le075c-dn -- Fri, 21 Sep 2007 13:31:21 -0400
 -->

</head>
<body BGCOLOR=white>
version executed successfully
<p>
<pre>
Lumenera/explorer2 r5281 hw1.1 1.8.0.10 : le075c-dn -- Fri,
21 Sep 2007 13:31:21 -0400

</pre>
<p>
<a href="/index.html">Home Page</a>
</body>
</html>
```

2.4.2.3 Response on Failure

If there is an error, `templates/error.tmpl` is processed and returns the command name and error number.

Copyright © 2009

Example: get a non-existent API property

```
http://CamIP/cgi-bin/get?foo
```

Failure template response:

```
<html>
<head><title>Error</title>
<!-- CAMERA ERROR 3 Invalid Argument property=foo: .
ctl.c:265 -->
</head>
<body>
get produced error:
<pre id=error>
Invalid Argument property=foo: . ctl.c:265
</pre>
</body>
</html>
```

Example: set a property to invalid type (string instead of double)

```
http://CamIP/cgi-bin/set?gain=X
```

Failure template response:

```
<html>
<head><title>Error</title>
<!-- CAMERA ERROR 3 Invalid Argument gain=X: invalid type.
ctl.c:215 -->
</head>
<body>
set produced error:
<pre id=error>
Invalid Argument gain=X: invalid type. ctl.c:215
</pre>
</body>
</html>
```

2.4.2.4 Access Levels

Lumenera Network cameras support three levels of access, as shown below.

Table 2-2 Available Access Levels in Network Cameras

Login	Password property in API	Privileges	CGI path	Webserver port in API
admin	admin_password	Full access	/cgi-bin	Port 80
user	user_password	Limited access, read-only	/cgi-usr	Port 80 and Port 8080
public	None	Very limited access	/cgi-pub	Port 80 and Port 8080

The `admin_password` sets access to the camera's `/cgi-bin` directory and `/admin.htm` file.

The `user_password` provides only the specific commands from the path `/cgi-usr` that are listed below:

```
/cgi-usr/get
/cgi-usr/image
/cgi-usr/nph-image
/cgi-usr/nph-image.jpg
/cgi-usr/nph-video
/cgi-usr/version
/cgi-usr/video
```

Where a request is made for admin access without proper authentication, the standard 401-Not Authorized response is issued:

```
<HTML><HEAD><TITLE>401 Unauthorized</TITLE></HEAD>
<BODY><H1>401 Unauthorized</H1>
Your client does not have permission to get URL /cgi-
bin/version from this server.
</BODY></HTML>
```

In the example above, a valid request at the user access level could specify `/cgi-usr/version` instead of `/cgi-bin/version`.

Public access allows access to the following commands:

```
/cgi-pub/
/version
/index.html
/images/
/javascript/
/favicon.ico
```


Copyright © 2009

2.4.3 Telnet Environment

The telnet client establishes a TCP connection typically on port 23. It can be used to establish a connection to the camera to change camera settings, run camera functions or get the camera status.

To access any function in the Camera API through a telnet session, use either the `luether` command or `l` for short. API functions and their individual arguments are separated by a space. Below are some examples on how the functions can be accessed through a telnet client.

To save the current camera settings to non-volatile memory:

```
luether save_settings
l save_settings
```

To get an image from the image buffer:

```
luether image number=-2
l image number=-2
```

To access a camera property, use the `get` or `set` API functions as shown below:

To get the current camera gamma value

```
luether get gamma
l get gamma
```

To set the camera's exposure and gain in one command

```
luether set exposure=20 gain=2.0
l set exposure=20 gain=2.0
```

3

API Function Reference

3.1 Summary of API Functions

Table 3-1 provides a quick summary of the camera API functions supported for the Network cameras. Not all functions listed below will be supported for every camera

Table 3-1 Summary of Camera API Functions

Function	Description
add_parameter	Adds a user-defined property
ftp_images	Transfers JPEG images to an FTP server
ftp_stream_last_error	Returns last error from asynchronous FTP stream
ftp_stream_status	Returns status of an asynchronous FTP stream
ftp_stream_stop	Stops a particular FTP stream
ftp_stream_stop_all	Stops all FTP streams
get	Returns the value of the specified property
get_settings	Gets a summary of property values
image	Transfers a single image in JPEG format
image_raw	Transfers a single image in RAW format
image_yuv	Transfers a single image in Abekas YUV format
lens_reset_aperture	Synchronizes the iris control of a Canon lens
load_settings	Loads camera settings from non-volatile memory
put_settings	Sets the values of multiple properties at once
put_settings_from_property	Applies settings stored in a property
reset	Reboots the camera
save_settings	Saves the current settings to non-volatile memory
set	Sets the specified property to a new value

Copyright © 2009

Function	Description
start	Starts the image processing pipeline
stop	Stops the image processing pipeline
strobe	Causes a single strobe pulse on the GPIO output
subimage_yuv	Transfers a sub-image in Abekas YUV format
test_email_alarm	Tests the motion email script
udp_stream	Transfers JPEG images to an UDP client
udp_stream_stop	Stops a particular UDP stream
udp_stream_stop_all	Stops all UDP streams
version	Returns firmware version information
video	Transfers a stream of JPEG images

3.2 API Function Description

This section describes all of the camera's API functions and how to use them.

add_parameter

This function allows the saving of user-defined properties on the camera. Once properties have been added to the camera, the `get` and `set` command can be used to access their values.

Arguments

`name` *string*
Must not contain spaces

`value` *string*
An arbitrary string

Return Values

Returns nothing on success. Throws a fault on error.

Notes

Setting the value of a property that does not exist results in an error. Use `save_settings` API function to save all user-defined properties created using `add_parameter` across camera reboots.

Example

```
CGI:
/cgi-bin/add_parameter?name=MyProperty&value=MyValue
Telnet:
l add_parameter name=MyProperty value=MyValue
```


Copyright © 2009

ftp_images

Transfers an image or a stream of JPEG images to an FTP server.

Arguments

`server` *string*
The server name or IP address

`path` *string*
Relative or absolute file path on the FTP server in which image(s) will be stored.

`user` *<optional> string*
FTP user name { default: ftp }

`password` *<optional> string*
FTP password { default: ftp }

`pre` *<optional> double*
Duration in seconds of the pre-trigger images to be transferred. { default: 0 }

`post` *<optional> double*
Duration in seconds of the post-trigger images to be transferred. { default: 0 }

`passive` *<optional> boolean*
FTP mode. If set to 1, passive FTP mode (PASV) is used. If set to 0, PORT mode is used. { default: 0 }

`async` *<optional> boolean*
Async FTP mode. If set to 1, the command returns immediately and the FTP streaming is executed in the background with errors sent to the camera's `syslog`. If set to 0, the command returns when the FTP streaming is complete with errors returned normally. { default: 0 }

`session` *<optional> string*
Session name. { default: noname }

`framerate` *<optional> double*
The frame rate of the desired FTP stream measured in frames per second (fps). If this value is larger than the camera's current frame rate, images will be transferred at the fastest rate possible. { default: 1000 }

`archive` *<optional> boolean*
Run in archive mode. If set to 1, the camera will use its rolling buffer to send images without dropping any frames, resulting in increased latency when network bandwidth is limited. If set to 0, the camera may drop frames rather than increase latency. { default: 1 }

`ref_date` *<optional> string*

Date and time for the requested index frame (image number 0) to a precision of seconds. The camera will send images beginning with the most recent image still in the buffer that was captured prior to the requested time. A request for images that are no longer present in the buffer will produce the most recent images possible. The `ref_date` format is the same as that obtained from the `/cgi-bin/get?date` or `l get date` commands (RFC 2822, see the documentation for the `date` property in the API). The `ref_date` must be URL encoded. If `ref_date` is used then `ref_useconds` is also required. The `ref_date` is ignored if it is empty (null).

`ref_useconds` *<optional> int*

The portion of the time of the requested index frame (image number 0) in microseconds. { default: -1, `ref_useconds` is ignored }

Return Values

Returns nothing on success. Throws a fault on error. Error can be retrieved by calling `ftp_stream_last_error` API function.

Notes

If `path` specifies directories that do not exist, the directories will be created. `path` is first run through `strftime` so that a date can be added to the path name. For full documentation of the conversion specifications, please consult `strftime` documentation from: http://www.opengroup.org/.

If `path` includes `%%i` (or any other legal Unix/SUSv2 printf variation such as `%%05i`) the relative image number will be inserted. Increasing numbers apply to successive images. The most recent image is assigned the number 0. Pre-trigger images are assigned negative numbers.

If `path` includes %%Q, the subsecond portion of the time in units of milliseconds [000-999] will be inserted.

An example path follows that uses the year, month [1..12], day of month [1..31], 24-hour time (hours, minutes, and seconds), and a relative image number.

Example path:

```
MyFile%Y%m%d%H%M%S%%05i.jpg
```

Example result:

```
MyFile20081231235959000000.jpg
```

If both `post` and `pre` values are zero then only one frame will be transferred. The maximum value for `pre` is related to `number_of_buffers` and the global `framerate` property. To stream indefinitely, set `post` to a large value such as

Copyright © 2009

1E+20. `ftp_stream_stop` commands will not work if `async=0`. To stop streaming when `async`=0, close the associated pipe.

Any of the arguments that contain special characters or spaces must be URL encoded. This usually applies only to the path. The password may also be affected. This is the reason for the use of `` `echo -n TEXT|url_escape` `` in the `path` argument in the Telnet example below.

Immediately after a change in the image geometry (`x`, `y`, `width`, and/or `height`), wait at least 2*(1/`framerate`) or a subsequent `ftp_images` call may fail with an error.

Examples

```
CGI: (use %25 URI escape code for the '%' character)
     ftp_images?server=10.10.111.111&path=image%25H%25M%25S
     %25%25Q.jpg&user=guest&password=guest&pre=10&post=10&a
     rchive=1&framerate=1

     ftp_images?server=10.10.111.111&path=x%25H%25M%25S%25%
     25Q.jpg&user=guest&password=guest&post=3&ref_useconds=
     0&ref_date=Fri%2C+11+Jan+2008+17%3A52%3A00+%2D0500%0A

Telnet:
     l ftp_images server=10.10.111.111 user=guest
     password=guest pre=10 post=10 archive=1 framerate=1
     path=`echo -n image%H%M%S%%Q.jpg|url_escape`

     l ftp_images server=10.10.111.111 user=guest
     password=guest ref_useconds=0 post=3
     path= x%25H%25M%25S%25%25Q.jpg
     ref_date=Fri%2C+11+Jan+2008+17%3A52%3A00+%2D0500%0A
```

ftp_stream_last_error

Returns the last error, if any, from an `ftp_images` stream. The stream does not need to be running to get this error.

Arguments

`session` *string*

Session name. For an alarm FTP session, use 'alarm' for this argument. For the default session name, use `noname`.

Return Values

Returns the last error as a string. Throws a fault if the command fails.

Typical error message returned are:

```
Unable to connect to [FTP server IP address]
Login failure
ftp open
```

Notes

Querying the last error does indicate if there was a subsequent successful FTP session using the same session name. For example, if the first session produces the error `Login failure` and is followed by another successful session that uses the same session name, then `ftp_stream_last_error` will still report `Login failure`.

The return value of `ftp_stream_last_error` may not reflect attempts to call the `ftp_images` function where a required argument cannot be validated and the FTP session is never attempted. For example, calling `ftp_images` with the `path` missing or incorrectly formatted fails with an error `Missing Argument path`, but this does not affect the return value of a later call of `ftp_stream_last_error`.

A return value of `ftp open` may indicate that a recent call to `ftp_images` conflicted with an earlier call with the same session name. For example, very frequent alarm FTP events when using a small value of the `alarm_delay` property may lead to conflicts. In particular, errors may result where more than one call to `ftp_images` attempts to write to the same file `path` on the same `server`.

Example

```
CGI:
     /cgi-bin/ftp_stream_last_error?session=alarm
Telnet:
     l ftp_stream_last_error session=alarm
```


Copyright © 2009

ftp_stream_status

Returns the status of the FTP stream.

Arguments

`session` *string*
Session name. For an alarm FTP session, use 'alarm' for this argument.

Return Values

Returns a boolean on success where a value of 1 states that the asynchronous stream is running. Throws a fault on error. Also returns a value of if the stream has produced an error, but will be retried. In this case, `ftp_stream_last_error` returns the value of the last error.

Notes

None.

Example

```
CGI:
     /cgi-bin/ftp_stream_status?session=alarm

Telnet:
     l ftp_stream_status session=alarm
```

Copyright © 2009

ftp_stream_stop

Stops an FTP stream.

Arguments

`session` <optional> *string*

Session name. The default session name is `noname`. For an alarm FTP session, use 'alarm' for this argument.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

None.

Example

```
CGI:
     /cgi-bin/ftp_stream_stop?session=alarm
or
     /cgi-bin/ftp_stream_stop?session=alarm

Telnet:
     l ftp_stream_stop session=alarm
```


Copyright © 2009

ftp_stream_stop_all

Stops all FTP streams.

Arguments

None.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

If you call `ftp_stream_stop_all`, calls to `ftp_images` started and still running in a Telnet session will exit with the message `Terminated` or `Killed` echoed on the console.

Example

```
CGI:
     /cgi-bin/ftp_stream_stop_all

Telnet:
     l ftp_stream_stop_all
```

get

Returns the value of a specified property.

Arguments

`property_name` *string*
Valid property name

Return Values

On success, returns the value of the property. Throws a fault on error.

Notes

None.

Examples

```
CGI:
     /cgi-bin/get?autogain
     // Return value 1 (true) or 0 (false)
or
     /cgi-bin/get?width
     // Returns an integer, the image width in pixels.

Telnet:
     l get autogain
     // returns 1 (true) or 0 (false)
or
     l get autoexposure
     // returns 1 (true) or 0 (false)
or
     l get gain; l get exposure; l get luminance;
     // returns all values in sequence
```


Copyright © 2009

get_settings

This command returns an array of property/value pairs representing all of the current settings in the camera. Each property/value pair is separated from the next pair by the user definable delimiter string.

Arguments

`delimeter` <optional> *string*

Separation character sequence that defines the individual property/value pair. Some settings may contain a carriage return/line feed (CRLF) sequence as part of their value. Setting the `delimeter` to a unique value other than a simple CRLF will ensure that parsing the response will be accurate. If a zero-length string is passed, the delimeter will be `\0` (NULL). { default: \r\n***\r\n }

Return Values

Array of property/value pairs on success; throws a fault on error.

Notes

The entire table of delimited settings is issued twice following the standard format of `templates/success.tmpl`, once in the `<--LUMENERA -->` or `<--CAMERA -->` tag and again in the body of the HTML. Only property values that are saved in the configuration database will be returned.

Example

```
CGI:
     /cgi-bin/get_settings

Telnet:
     l get_settings
```

image

Transfers a single image in JPEG format.

Arguments

`number` *<optional> int*

Specifies the image in the rolling buffer to be returned where `1` corresponds to the most recent image in the buffer. `number_of_buffers-1` indicates the oldest image. A value of `0` requests the image currently being integrated. Note that it is possible to pass a value of number that is out of range based on the current value of `number_of_buffers`. If `ref_date` is used then `ref_useconds` is also required. The `ref_date` is ignored if it is empty (null). { default: 1}

`ref_date` *<optional> string*

Date and time for the requested index frame (image number 0) to a precision of seconds. The camera will send images beginning with the most recent image still in the buffer that was captured prior to the requested time. A request for images that are no longer present in the buffer will produce the most recent images possible. The `ref_date` format is the same as that obtained from the `/cgi-bin/get?date` or `l get date` commands (RFC 2822, see the documentation for the `date` property in the API). The `ref_date` must be URL encoded. If `ref_date` is used then `ref_useconds` is also required. The `ref_date` is ignored if it is empty (null).

`ref_useconds` *<optional> int*

The portion of the time of the requested index frame (image number 0) in microseconds [0 to 999999].. { default: -1, `ref_useconds` is ignored}

Return Values

If the camera is stopped, a value of `0` for the `number` may cause this command to hang for an undefined period of time. Throws a fault on error. Returns a binary JPEG image file on success. The format of the response is as follows.

```
HTTP/1.0 200 OK
Server: lumenera nph-cgi
Cache-Control: no-store
Pragma: no-store
Content-Length: 443552
Content-Type: image/jpeg
```
[EXIF header and JPEG image data follows]

Copyright © 2009

Notes

Use nph-image or nph-image.jpg to make a No Process Header request for reduced latency.

Immediately after a change in the image geometry (`x`, `y`, `width`, and/or `height`), wait at least 2*(1/`framerate`) or a subsequent `image` call may fail with an error.

Example

```
CGI:
     /cgi-bin/nph-image.jpg

Telnet:
     l image
```

load_settings

Loads camera settings from a file in non-volatile memory.

Arguments

`filename` `<optional> string`

Filename of saved settings file. { default: persistent }

Return Values

Returns nothing on success. Throws a fault on error.

Notes

The filename can be any arbitrary name that was created using `save_settings` API function or one of two special filenames: `persistent` or `factory_defaults`. The filename `persistent` specifies the current power-on defaults. The filename `factory_defaults` specifies the default settings with which the camera shipped.

Example

```
CGI:
     /cgi-bin/load_settings

Telnet:
     l load_settings
```


Copyright © 2009

ptz_command

Performs a pan-tilt-zoom (PTZ) control action of a motorized positioning system and motorized lens.

Arguments

`command` *<optional> string*

The action to perform, where command is one of the following values. {default: stop}

Pan-Tilt actions: stop, up, up_left, up_right, down, down_left, down_right, left, right.

Lens focus actions: far, near.

Lens iris actions: open, close.

Lens zoom actions: tele, wide.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

Applies only to Pan Tilt Zoom (PTZ) systems that use the camera's RS232 serial port to interface the network camera to a pan-tilt motor and a motorized zoom lens. Calling `ptz_command` sends a command to a PTZ controller board via the camera's RS232 serial port.

Each action is performed in one pre-defined step size. The same actions and step sizes are used in the graphical web page user interface for PTZ control.

Set `ptz_enable` to be true when using `ptz_command`. A system error is thrown unless `serial_use`=accessory and `serial_speed`=2400.

Example

```
CGI:
     /cgi-bin/ptz_command?up_left

Telnet:
     l ptz_command up_left
```

ptz_preset

Used to instruct the camera to set, clear, or move to a particular pan-tilt-zoom (PTZ) camera preset position.

Arguments

`command` *<required> string*

The action to take: set, goto, or clear.

set: Set the specified PTZ preset `id_number` at the current position.
goto: Move to the PTZ preset position associated with `id_number`.
clear: Clear the PTZ preset associated with `id_number`.

`id_number` *<required> string*

The preset number to which the action applies.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

Applies only to Pan Tilt Zoom (PTZ) systems that use the camera's RS232 serial port to interface the network camera to a pan-tilt motor and a motorized zoom lens. Calling `ptz_preset` sends a command to a PTZ controller board via the camera's RS232 serial port.

Set `ptz_enable` to be true when using `ptz_command`. A system error is thrown unless `serial_use`=accessory and `serial_speed`=2400.

Example

```
CGI:
     /cgi-bin/ptz_preset?set&3
     /cgi-bin/ptz_preset?goto&3
     /cgi-bin/ptz_preset?clear&3
Telnet:
     l ptz_preset set 3
     l ptz_preset goto 3
     l ptz_preset clear 3
```


Copyright © 2009

put_settings

Sets multiple properties in a single function call. This function has the same effect as calling the `set` API function multiple times for each property or with multiple property-value pairs. Each property-value pair must be separated from the next pair by the user-defined delimiter. The first equals sign in a name-value pair separates the name from the value. A value can contain equals signs (=).

Arguments

`settings` *string*
property/value pairs string.

`delimiter` *<optional> string*
Delimiter that separates the property/value pair. { default: “,” }

Return Values

Returns nothing on success. Throws a fault on error.

Notes

The delimiter cannot be \0 (NULL). It is recommended that the delimiter not contain a carriage return/line feed pair (CRLF) as text translation may occur. If using the CGI interface or the command line, standard URI escapes are used.

Example

```
CGI:
     /cgi-
     bin/put_settings?gain%3D1%2Cexposure%3D3&delimeter=%2C

Telnet:
     l put_settings gain%3D1%2Cexposure%3D2 delimeter=%2C
```

put_settings_from_property

Sets multiple properties from a given property. Similar to the `put_settings` API function, this function sets multiple properties in a single function call. It retrieves the value from an existing property and applies it to multiple properties.

Arguments

`property_name` *string*
valid existing property name

`delimiter` *<optional> string*
Delimiter that separates the property/value pair. { default: “,” }

Return Values

Returns nothing on success. Throws a fault on error.

Notes

The delimiter cannot be \0 (NULL). If the current camera settings are unwanted, the command

```
put_settings_from_property default_image_params
```

can be used to restore the Image Defaults settings. This command is similar to

```
load_settings factory_defaults
```

with the exception that none of the network settings are affected.

Example

```
CGI:
     /cgi-bin/put_settings_from_property?
     default_image_params

Telnet:
     l put_settings_from_property default_image_params
```


Copyright © 2009

reset

Reboots the camera.

Arguments

None.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

The camera will be unresponsive for approximately 30 seconds until it has finished booting.

Example

```
CGI:
     /cgi-bin/reset

Telnet:
     l reset
```

save_settings

Saves the current camera settings to non-volatile memory. The settings may be saved to a user-defined filename so that they may be reloaded via **load_settings**, or the default filename 'persistent' may be used. The special filename 'persistent' defines the set that will be automatically loaded by the camera when rebooted.

Arguments

`filename` *<optional> string*
Filename for saved settings file. { default: persistent }

Return Values

Returns nothing on success. Throws a fault on error.

Notes

None.

Example

```
CGI:
     /cgi-bin/save_settings

Telnet:
     l save_settings
```


Copyright © 2009

set

Set the value of a particular property.

Arguments

`property` *string*
Existing property name

`value` *<based on property>*
Value of the same data type as the property.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

None.

Examples

```
CGI:
     \cgi-bin\set?autogain=1
or
     \cgi-bin\set?width=300&height=200&x=50&y=50

Telnet:
     l set autogain=1
or
     l set autoexposure=1
or
     l set width=300 height=200 x=50 y=50
```

start

Starts the image processing pipeline on the camera.

Arguments

None.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

The `start` command is disruptive to normal camera operations. Use of the `start` command is not recommended.

Example

```
CGI:
     /cgi-bin/start

Telnet:
     l start
```


Copyright © 2009

stop

Stops the image processing pipeline in the camera.

Arguments

None.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

The `stop` command is disruptive to normal camera operations. Use of the `stop` command is NOT recommended.

Example

```
CGI:
     /cgi-bin/stop

Telnet:
     l stop
```

strobe

Causes a single analog pulse to occur on the GPIO output.

Arguments

None.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

The property `output_select` must be set to 'on' for this function to work correctly. The duration of the pulse is set using the `strobe_length` property.

Example

```
CGI:
     /cgi-bin/strobe

Telnet:
     l strobe
```


Copyright © 2009

test_email_alarm

Unconditionally execute the `motion_email_script`. This will force a motion alarm type email message to be generated and sent to the assigned email address.

Arguments

None.

Return Values

Returns an int value on success. Throws a fault on error.

Notes

See documentation for the `motion_email_script` property. You must specify a valid mail server and email recipient.

Example

```
CGI:
     /cgi-bin/
     set?motion_email_recipient=me%40mydomain%2Ecom
     /cgi-bin/set?motion_email_server=123%2E123%2E123%2E123
     /cgi-bin/test_email_alarm

Telnet:
     l set motion_email_recipient=me@mydomain.com
     l set motion_email_server=123.123.123.123
     l test_email_alarm
```

udp_stream

Transfers an image or a stream of JPEG images to a UDP client.

Arguments

`client` *string*
Client name or IP address.

`port` *int*
Client port address.

`end` *<optional> int*
The image number for the last image to transfer from the rolling buffer. If `start` is 0, this value corresponds to the number of images to transfer. If `end` is less than `start`, then the UDP streaming will continue indefinitely. { default: 1 }

`start` *<optional> int*
The first image to transfer from the rolling buffer. A positive number refers to future images not yet captured, a negative number refers to older images in the rolling buffer. { default: 0 }

`frameskip` *<optional> double* =(0-n; default: 0)
The number of frames to discard for every frame sent. Fractional values are supported. This is in addition to skips required by the camera's `maximum_framerate` global value. { default: 0 }

`framerate` *<optional> double*
The desired frame rate of the UDP stream. If this value is non-zero, the camera calculates an appropriate `frameskip` value. If the global `framerate` value changes, the frame rate of this stream will change proportionately. { default: 0 }

`archive` <optional> boolean
Run in archive mode. If set to 1, the camera will use its rolling buffer to send images without dropping any frames, resulting in increased latency when network bandwidth is limited. If set to 0, the camera may drop frames rather than increase latency. { default: 0 }

Return Values

Returns nothing on success. Throws a fault on error.

Notes

The first 16 bytes for each packet in the UDP stream are four integers, each 4 bytes long, which are described below.

Magic number: Ensures the data received is from the camera.

Copyright © 2009

Image sequence number: Specifies the frame number from the camera.

Packet position/image offset: Location of this packet in the image in bytes. For example, a packet position of X means this packet should be inserted into the image at the Xth byte of the image.

Total image length in bytes: This should be the same for each packet for a given image.

The remaining bytes in the packet are image data. Each packet should be less than 64KB in size.

Immediately after a change in the image geometry (`x`, `y`, `width`, and/or `height`), wait at least 2*(1/`framerate`) or a subsequent `udp_stream` call may fail with an error.

Example

```
CGI:
     /cgi-bin/udp_stream?client=123.123.123.123&port=1234

Telnet:
     l udp_stream client=123.123.123.123 port=1234
```

udp_stream_stop

Stops a UDP stream.

Arguments

`client` string
The client name or address.

`port` int
The client port.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

None.

Example

```
CGI:
     /cgi-bin/udp_stream_stop?client=123.123.123.123&port=
     1234

Telnet:
     l udp_stream_stop client=123.123.123.123 port=1234
```


Copyright © 2009

udp_stream_stop_all

Stops all open UDP streams.

Arguments

None.

Return Values

Returns nothing on success. Throws a fault on error.

Notes

None.

Examples

```
CGI:
     /cgi-bin/udp_stream_stop_all

Telnet:
     l udp_stream_stop_all
```

version

Returns a string describing the camera hardware and software. This includes the hardware version, camera name, firmware version, and firmware build date.

Arguments

None.

Return Values

Returns a string on success. Throws a fault on error.

Notes

None.

Examples

```
CGI:
     /cgi-bin/version

Telnet:
     l version
```

Sample Output

```
Lumenera/explorer2 r5565 hw1.2 1.8.0.16 : le165c-dn -- Mon,
24 Dec 2007 12:31:56 -0500
```

`Lumenera/explorer2` Indicates the camera type

`r5565` Firmware build number

`hw1.2` Hardware version, “hw1.3” indicates extended RAM

`1.8.0.16` Firmware version
`1.8` is the major version of firmware.
`.0` Firmware branch
(number = main branch, A=alpha firmware, B=beta firmware)
`.16` Minor version, with even/odd value for release type
(even digit = production release, odd digit = release candidate)

`le165c-dn` Camera model

`Mon, 24 Dec 2007 12:31:56 -0500` Date and time of the firmware build

Copyright © 2009

video

Transfers a stream of images in JPEG format.

Optional Arguments

`type` <optional> string

Stream type. Two options available - `multipart/mixed` is used for older versions of the firmware, `multipart/x-mixed-replace` is used for newer versions and is more compatible with third party software. { default: multipart/mixed }

`frameskip` *<optional> double* =(0-n; default: 0)

The number of frames to discard for every frame sent. Fractional values are supported. This is in addition to skips required by the camera's `maximum_framerate` global value. { default: 0 }

`framerate` *<optional> double*

The desired frame rate of the UDP stream. If this value is non-zero, the camera calculates an appropriate `frameskip` value. If the global `framerate` value changes, the frame rate of this stream will change proportionately. { default: 0 }

`archive` <optional> boolean

Run in archive mode. If set to 1, the camera will use its rolling buffer to send images without dropping any frames, resulting in increased latency when network bandwidth is limited. If set to 0, the camera may drop frames rather than increase latency. { default: 0 }

`start` *<optional> int*

The first image to transfer from the rolling buffer. A positive number refers to future images not yet captured, a negative number refers to older images in the rolling buffer. { default: 0 }

`ref_date` *<optional> string*

Date and time for the requested index frame (image number 0) to a precision of seconds, formatted according to the requirements in RFC 2822. The camera will send the most recent buffered image that is prior to the requested time. Attempting to retrieve images outside the rolling buffer will send no images at all, even if the time specified is in the future or some of the images specified are in the buffer.

`ref_useconds` *<optional> int*

The portion of the time of the requested index frame (image number 0) in microseconds. { default: -1 }

Copyright © 2009

Return Values

Returns a binary JPEG image data on success. Throws a fault on error.

Notes

Local storage of Motion JPEG binary data in the camera file system is not recommended. The Telnet example that follows is for test purposes only.

Immediately after a change in the image geometry (`x`, `y`, `width`, and/or `height`), wait at least 2*(1/`framerate`) or a subsequent `video` call may fail with an error.

Examples

```
CGI:
     /cgi-bin/nph-video
or
     /cgi-bin/nph-video?type=multipart/x-mixed-replace&
     framerate=5
or
     /cgi-bin/nph-video?archive=1&start=-100

Telnet:
     Examples store binary data in camera file system and
     may exceed available storage capacity. For test
     purposes only. The video function is intended to be
     executed by a client computer.

     /cgi-bin/nph-video|cat > myfile.dat
     or
     l nph-video type=multipart/x-mixed-replace|cat >
     myfile.dat
     or
     l nph-video archive=1 start=-100|cat > myfile.dat
```

In all Telnet examples above use CTRL-C to halt their execution.

Copyright © 2009

4

API Property Reference

4.1 API Property Description

This section describes all of the camera's API properties and how to use them.

admin_password

Type

string

Description

Access to the administrator password for the camera. This property can be used to set the password.

Notes

If this property is not empty, a password is needed to access any of the camera configuration commands in the /cgi-bin/ directory.

The expected username is `admin`. The `admin` username and password may also be used to access the user sections of the camera.

To remove a password, set it to the null (empty) string.

If the password starts with "1", the `string` is assumed to be the MD5 hash of the password rather than the password itself.

Getting this property will return either `1` if a password is set or `null` otherwise.

alarm_date_last

Type

string, read-only

Description

The date and time of the last alarm event in ISO 8601 format. If no alarm has occurred since the camera's last boot, the value will be "none".

Note

Not supported by the Le045 and the Li series.

alarm_delay

Type

int

Description

Defines the delay time between alarms in seconds. Alarm notifications will not occur more frequently than once every `alarm_delay` seconds.

Notes

For the Le series cameras, `alarm_delay` is the time in seconds between alarms due to a signal on the general-purpose dry-contact input or a motion alarm defined using the `motion_map`.

For the Li series intelligent cameras, `alarm_delay` only affects events detected on the general-purpose dry-contact input. Alarms due to video analytics events are not affected by `alarm_delay` because such events are distinct and highly specific.

alarm_enable_email

Type

boolean

Description

Enable and disable alarms notifications via email. Alarm events can result from general-purpose hardware input, a motion alarm defined using the `motion_map` (Le series cameras), or a video analytics alarm event (Li series cameras).

Copyright © 2009

alarm_enable_ftp

Type

boolean

Description

Enable and disable alarm notifications via FTP. Alarm events can result from general-purpose hardware input, a motion alarm defined using the `motion_map` (Le series cameras), or a video analytics alarm event (Li series cameras).

alarm_enable_udp

Type

boolean

Description

Enable and disable alarm notifications via UDP. Alarm events can result from general-purpose hardware input, a motion alarm defined using the `motion_map` (Le series cameras), or a video analytics alarm event (Li series cameras).

Notes

When enabled and an alarm occurs, a UDP packet is sent to the client defined by `alarm_udp_client` on the `alarm_udp_port` port. The packet is in human readable format using carriage return/line feed pair as the separator:

`event number`: an incrementing unsigned long 32 bit integer

`IP address`: the camera's IP address (xxx.xxx.xxx.xxx)

`hardware address`: the camera's hardware MAC address (`xx:xx:xx:xx:xx:xx`)

`trigger time`: the time the trigger occurred (Tue Jun 10 12:09:46 2006)

`trigger reason`: one of

- `motion-centroid x:%d y:%d area:%2.4f` that specifies the X, Y coordinates where the motion was detected in the case of a motion alarm.
- `I/O input triggered` in the case of a dry-contact alarm trigger signal.

To acknowledge the alarm packet, send a UDP packet to the source address and port of the alarm packet. If not acknowledged, the alarm packet will be resent at one second intervals, to a maximum of three packets. The format of the acknowledgment is as follows::

```
event number\r\nACK\r\n
```

alarm_ftp_ramerate

Type

double

Description

FTP stream frame rate to use when an alarm is triggered.

Note

If this property is larger than the camera's current frame rate, images will be transferred as fast as possible assuming sufficient network bandwidth is available.

alarm_ftp_passive

Type

boolean

Description

Sets the FTP mode used by the camera to stream images to the FTP server on an alarm event. If true, the camera uses passive (PASV) mode FTP.

alarm_ftp_password

Type

string

Description

Password of the `alarm_ftp_user` account that will receive files via FTP on an alarm event.

alarm_ftp_path

Type

string

Copyright © 2009

Description

File path for FTP file transfer upon an alarm event.

alarm_ftp_post_trigger

Type

double

Description

Duration in seconds of images occurring after an alarm event that will be transferred after an alarm event occurs.

alarm_ftp_pre_trigger

Type

double

Description

Duration in seconds of images occurring before an alarm event that will be transferred when an alarm event occurs.

Note

Since new images are always being added to the rolling image buffer while the oldest images are overwritten, the maximum value for this property is related to `number_of_buffers` property and the camera's global frame rate.

alarm_ftp_server

Type

string

Description

IP address of the target FTP server that will receive files upon an alarm event.

alarm_ftp_user

Type

string

Description

User account name on the `alarm_ftp_server` FTP server that receives image files sent upon an alarm event.

alarm_udp_client

Type

string

Description

A comma-separated list of the target client IP addresses that receive UDP alarm packets.

Note

If the broadcast address for the subnet is specified, alarms will be broadcast. Each client address can include an optional appended port number, delimited by a colon. Errors appear in the camera file system in /var/log/messages.

alarm_udp_port

Type

int

Description

UDP client port that receives the UDP packets upon an alarm event.

Note

Errors appear in /var/log/messages. The port can also be specified in `alarm_udp_client` property string by separating it from the IP address using a colon.

analytics_alarm_enable

Type

boolean

Description

Enable or disable analytics alarm delivery. This parameter only has an effect when the camera is equipped with analytics algorithms (e.g. ObjectVideo) and

Copyright © 2009

those algorithms are enabled using `analytics_enable`. When true, the on-camera alarm responses to analytics alerts are enabled: email, FTP image transfer, UDP alarm packet, and general-purpose output.

Note

Applies only to the Intelligent (Li series) cameras.

analytics_email_script

Type

string

Description

Defines the script that is executed when an object video alarm is detected. The default value of this script uses the additional parameters motion_email_subject, motion_email_server, motion_email_recipient and alarm_delay to create and send an email message when object video alarm is detected. One needs to attach the object video alarm details received as the email body explicitly. See also motion_email_script.

Note

Applies only to the Intelligent (Li series) cameras.

analytics_enable

Type

boolean

Description

Enable or disable analytics.

Note

Applies only to the Intelligent (Li series) cameras.

analytics_view_select_model

Type

int

Description

This parameter defines the view model to use for the analytics algorithm.

Note

Possible values include:

0 Auto-Acquire Mode: This is the default value. When set, the analytics algorithm will take the current scene as the view for event detection. The algorithm will automatically adjust to a new scene as the view if/when the scene changes. A few frames are required for the algorithm to adapt to the new scene.

1 Auto-Force Mode: When camera starts up, the algorithm will use the first good scene as the view and will never change regardless if the scene changes in the future.

2 User Select Mode: When camera starts up, the algorithm will use the first good scene as the view. If the scene changes in the future, the algorithm will report the “unknown” view. It is up to the user to force the new scene as the new view in the analytics algorithm backend software.

Note

Applies only to the Intelligent (Li series) cameras.

aperture

Type

double

Description

The aperture of the lens expressed as an f-number.

Notes

This property applies only to camera models that use a Canon 35 mm lens.

A smaller f-number represents a more open iris.

When `iris_type` is set to `calibrated`, the `aperture` property value is limited by `aperture_min` and `aperture_max`. If the lens is not properly installed, `aperture` will return the value -1.

When `iris_type` is set to `none` or `dc`, `aperture` will return the value -1.

Copyright © 2009

aperture_hw_max

Type

double, read only

Description

The hardware-imposed upper limit on the `aperture` property, used to accommodate different lens types and is expressed as an f-number.

Notes

This property applies only to camera models that use a Canon 35 mm lens with `iris_type` property set to `calibrated`.

A smaller f-number represents a more open iris.

aperture_hw_min

Type

double, read only

Description

The hardware-imposed lower limit on the `aperture` property, used to accommodate different lens types and is expressed as an f-number.

Notes

This property applies only to camera models that use a Canon 35 mm lens with `iris_type` property set to `calibrated`.

A smaller f-number represents a more open iris.

aperture_max

Type

double

Description

The user-defined upper limit on the `aperture` property. This property can be read or set to a maximum aperture of the iris and is expressed as an f-number.

Notes

This property applies only to camera models that use a Canon 35 mm lens.

A smaller f-number represents a more open iris.

aperture_min

Type

double

Description

The user-defined lower limit on the `aperture` property. This property can be read or set to a minimum aperture of the iris and is expressed as an f-number.

Note

This property applies only to camera models that use a Canon 35 mm lens.

A smaller f-number represents a more open iris.

auto_algorithm

Type

string

Description

Specifies the method or algorithm used to calculate `luminance` for purposes of the auto brightness control (ABC).

Notes

Setting `autoexposure` and/or `autogain` to true will cause the camera to adjust the `exposure` time and/or `gain` to achieve the target luminance specified by `autoexposure_target`. See `auto_window_enable`, `auto_width`, `auto_height`, `auto_x`, and `auto_y` for information on defining the auto control window used for these calculations.

If `auto_algorithm=average`, the `luminance` value is calculated as the mean of the entire auto control window.

If `auto_algorithm=median`, the `luminance` value is calculated as the median of the whole auto control window. The median algorithm is less sensitive to extreme bright and dark areas than the `average` algorithm.

Copyright © 2009

If `auto_algorithm=peak`, the `luminance` value is calculated using only the brightest areas of the auto control window.

If the `auto_algorithm=dark`, the `luminance` value is calculated using only the darkest areas of the auto control window.

Applies only to the Intelligent (Li series) cameras.

auto_height

Type

int

Description

Height of the auto control window.

Notes

The auto control window defines the area that is used by the `auto_algorithm` method when applying `autoexposure`, `autogain` and `whitebalance`.

The auto control window is defined relative to the top left of the current viewing window that is defined by `height`, `width`, `x`, and `y`.

This property is enabled and disabled with `auto_window_enable` property.

auto_jpeg_quality

Type

Boolean

Description

Enables or disables the auto-JPEG quality option to maintain JPEG image file sizes. When enabled, this property automatically adjusts the JPEG quality so that the image file size in bytes is always between `max_jpeg_size` and `min_jpeg_size`.

Notes

For Le series cameras only, setting `auto_jpeg_quality` true will ensure that the file size is also kept below the `projected_network_bandwidth` value divided by the lesser of `projected_network_framerate` value or `framerate` value.

For the Le045 and the Intelligent (Li series) cameras, the `projected_network_bandwidth` and `projected_network_framerate` properties are not supported. See `target_jpeg_size`.

auto_width

Type

int

Description

Width of the auto control window.

Notes

The auto control window defines the area that is used by the `auto_algorithm` method when applying `autoexposure`, `autogain` and `whitebalance`.

The auto control window is defined relative to the top left of the current viewing window that is defined by `height`, `width`, `x`, and `y`.

This property is enabled and disabled with `auto_window_enable` property.

auto_window_enable

Type

boolean

Description

Enables or disables the use of a ROI for calculating the value for the `auto_algorithm`.

Notes

When set, the `auto_x`, `auto_y`, `auto_width` and `auto_height` properties are used to define the region of interest (ROI) that is used by the `auto_algorithm` to calculate the `luminance` and `whitebalance` values. The auto control window is defined relative to the top left of the current viewing window that is defined by `height`, `width`, `x`, and `y`. The auto-brightness control also requires `autogain`, `autoexposure`, and/or `autoiris` to be true.

When cleared, the entire viewing window is used for the `auto_algorithm` calculations.

Copyright © 2009

auto_x

Type

int

Description

Horizontal offset of the auto control window from the left edge of the current viewing window.

Notes

The auto control window defines the area that is used by the `auto_algorithm` method when applying `autoexposure`, `autogain` and `whitebalance`.

The auto control window is defined relative to the current viewing window. This window is defined by `height`, `width`, `x`, and `y`.

This property is enabled and disabled with `auto_window_enable` property.

auto_y

Type

int

Description

Vertical offset of the auto control window from the top edge of the current viewing window.

Notes

The auto control window defines the area that is used by the `auto_algorithm` method when applying `autoexposure`, `autogain` and `whitebalance`.

The auto control window is defined relative to the current viewing window. This window is defined by `height`, `width`, `x`, and `y`.

This property is enabled and disabled with `auto_window_enable` property.

autoexposure

Type

boolean

Description

Enable or disable the `autoexposure` functionality.

Notes

When disabled, the exposure time is strictly controlled by the value of the `exposure` property. When enabled, the camera adjusts the exposure to match the requested luminance set by the `autoexposure_target` value using the currently specified `auto_algorithm` algorithm. The camera also uses the `lighting_frequency` property to limit the exposure value to predefined multiples in order to avoid banding or flickering in images when using AC powered light sources.

The `auto_algorithm`'s primary guidance is to minimize noise in the image so it prefers to lower the `gain` rather than lower `exposure`, and it prefers to raise `exposure` rather than raise `gain`.

Not supported by Le045 and Intelligent (Li series) cameras.

autoexposure_knee

Type

double

Description

Defines an exposure value where the `auto_algorithm` will stop adjusting the exposure and starts adjusting the gain value until both the `autoexposure_knee` and `autogain_knee` points are reached.

Notes

If both `autoexposure` and `autogain` are set, the camera will achieve the `autoexposure_target` by adjusting either `exposure` or `gain`. The `auto_algorithm` will adjust one property until it reaches the knee point value and then adjust the other property to achieve the desire `luminance` value. Once both values are reached, the exposure and gain values are adjusted to the respective limits.

The `auto_algorithm`'s primary guidance is to minimize noise in the image so it prefers to lower the `gain` rather than lower `exposure`, and it prefers to raise `exposure` rather than raise `gain`.

The setting `autoexposure_knee=0` corresponds to setting this value to the maximum exposure that does not slow down the camera's frame rate.

Copyright © 2009

autoexposure_target

Type

double

Description

The `luminance` target. The camera will adjust its settings in order to achieve an image with `luminance` near the value provided.

Notes

The `luminance` is calculated as specified by the `auto_algorithm` property.

If `autoiris`, `autoexposure`, and/or `autogain` are enabled, the camera will adjust the iris diameter (`aperture`), `exposure` and/or `gain`, respectively.

Not supported by Le045 and Li045 cameras.

autogain

Type

boolean

Description

Enable or disable `autogain` functionality.

Notes

When disabled, the gain value is strictly controlled by the value of the `gain` property. When enabled, the camera adjusts the gain to match the requested luminance set by the `autoexposure_target` value using the currently specified `auto_algorithm` algorithm.

The `auto_algorithm`'s primary guidance is to minimize noise in the image so it prefers to lower the `gain` rather than lower `exposure`, and it prefers to raise `exposure` rather than raise `gain`.

autogain_knee

Type

double

Description

Defines a gain value where the `auto_algorithm` will stop adjusting the gain and starts adjusting the exposure value until both the `autoexposure_knee` and `autogain_knee` points are reached.

Notes

If both `autoexposure` and `autogain` are set, the camera will achieve the `autoexposure_target` by adjusting either `exposure` or `gain`. The `auto_algorithm` will adjust one property until it reaches the knee point value and then adjust the other property to achieve the desire `luminance` value. Once both values are reached, the exposure and gain values are adjusted to the respective limits.

The `auto_algorithm`'s primary guidance is to minimize noise in the image so it prefers to lower the `gain` rather than lower `exposure`, and it prefers to raise `exposure` rather than raise `gain`.

autoiris

Type

boolean

Description

Enable or disable the auto iris functionality.

Notes

When set and when the `exposure` value begins to drop below the `minimum_exposure` value, the camera closes the lens iris to achieve the requested `luminance` specified by `autoexposure_target`. If the target `luminance` cannot be achieved due to low light conditions, the `autoexposure` and/or `autogain` functionality is activated when enabled.

boot_version

Type

string, read-only

Description

Returns a string describing the bootloader version.

Copyright © 2009

brightness

Type

double

Description

Adjusts the brightness value of the camera. This value can be between -256 (black) and +256 (white).

Note

Not supported by Le045 and Li series cameras.

color_conversion

Type

string

Description

Specifies the RGB->YCbCr color conversion matrix to be applied after color gains and `whitebalance` are applied. Available values are `yuv`, `709` (ITU-R BT.709 specification), and `601` (ITU-R BT.601.5 specification).

color_preset

Type

string

Description

Specifies the color correction matrix to be applied after color gains and `whitebalance` are applied. Available values are `unity`, `default`, `chip_default`, `fluorescent`, `daylight`, `xenon_flash` and `halogen`.

contrast

Type

double

Description

Accesses the contrast value of the camera. It can take values between -100 and +100.

crosshair_enable

Type

boolean

Description

Enable or disable crosshair overlay onto images captured from the camera. This feature is used temporarily when setting up the camera.

Notes

The `privacy_map` may need to be disabled temporarily in order for the alignment crosshairs to be visible.

Not supported by Le045 and Intelligent (Li series) cameras.

current_broadcast_address

Type

string, read-only

Description

Provides the camera's current broadcast address.

current_fqdn

Type

string, read-only

Description

Provides the fully qualified domain name.

Note

If both `name` and `dns_domain_name` properties are set and the camera has been rebooted so that these properties have taken effect, then this value is

Copyright © 2009

simply `name.dns_domain_name`. Otherwise, it is obtained by asking the associated `nameserver`.

current_gateway

Type

string, read-only

Description

Provides the gateway used by the camera.

current_ip

Type

string, read-only

Description

Provides current IP address assigned to the camera.

current_nameserver

Type

string, read-only

Description

Provides the nameserver assigned to the camera.

current_netmask

Type

string, read-only

Description

Provides the IP subnet mask assigned to the camera.

date

Type

string

Description

Accesses the camera's current date in RFC 2822 format.

Example: "Mon, 10 Nov 2006 15:48:06 -0500".

date_utc

Type

string, read-only

Description

Provides the camera's current UTC date in RFC 2822 format.

Example: "Mon, 10 Nov 2006 15:48:06 +0000".

day_mode

Type

boolean

Description

Used to set or query the day-night mode of the camera.

If `day_mode`=1 (true, the Factory Default/Image Defaults/startup value), the camera is in day mode and uses the settings defined in the property `normal_light_settings`.

If `day_mode`=0 (false), the camera is in night mode and uses the settings defined in the property `low_light_settings`.

Notes

Applies only to the Le series cameras. Requires firmware 1.8.1.14 or higher.

Does not apply to the Le045 cameras.

The day-night mode can be set by the user (e.g "`l set day_mode=0`") to force a mode change.

Copyright © 2009

Automatic day-night settings changes can be set up that use the thresholds set in `low_light_criteria` and `normal_light_criteria`, if present. Such settings can be defined for cameras with or without a day-night (visible/near-infrared) filter shuttle option. For camera models with the “DN” suffix to the model number, a night mode with black and white images (`saturation`=-100) and an infrared-transparent glass in the light path of the camera (`ir_shuttle_position`=1) is the factory default configuration.

`day_mode` can be saved/loaded just like any other parameter.

`day_mode` will be reset to factory default value (1, true) by clicking "Image Defaults".

daynight_bw_mode

Type

boolean

Description

If true (default), the camera will disable color during night mode operation.

Note

Applies only to the Le045 and Li045 cameras.

daynight_gain_boost

Type

boolean

Description

If true, the camera will boost gain by 6 dB during night mode operation. Default is false (gain boost disabled),

Note

Applies only to the Le045 and Li045 cameras.

daynight_gain_in_threshold

Type

double

Description

The gain level at which the camera enters into night mode when `daynight_mode` is set to `auto`. Default is 16.

Notes

Must be greater than `daynight_gain_out_threshold`.

Takes effect only if `autogain` is enabled and `daynight_mode` is `auto`.

Applies only to the Le045 and Li045 cameras.

daynight_gain_limit

Type

double

Description

Determines the highest gain that will be applied in night mode. Default is 31.

Notes

Applies to all `daynight_mode` settings (`auto`, `external`, and `on`).

Applies only to the Le045 and Li045 cameras.

daynight_gain_out_threshold

Type

double

Description

The gain level at which the camera exits night mode. Default is 8.

Notes

Must be less than `daynight_gain_in_threshold`.

Takes effect only if `autogain` is enabled and `daynight_mode` is `auto`.

Applies only to the Le045 and Li045 cameras.

Lumenera corporation
Copyright © 2009

daynight_mode

Type

string

Description

Default is `auto`. Legal values are:

- `off` (always in "day" mode);
- `on` (always in "night" mode), and
- `auto` (switch in and out of "night" mode depending on the system `gain`, `daynight_gain_out_threshold` and `daynight_gain_in_threshold`.

Notes

Applies only to the Le045 and Li045 cameras.

default_image_params

Type

string

Description

Provides a list of default image parameters that can be used by `put_settings_from_property` API function to set the camera image properties back to the factory default state without affecting the non-image properties.

description

Type

string

Description

Provides a string describing the camera. By default, the contents describe the camera model and MAC address. This property can be changed to provide a more descriptive string for a specific application.

Copyright © 2009

dhcp_fallback

Type

string

Description

Provides the default IP address fallback when no address information is provided when the camera is enabled with DHCP. The factory setting for `dhcp_fallback` is 192.168.1.222.

Note

If this value is empty, and DHCP times out, then no IP address will be set.

dhcp_timeout

Type

int

Description

Timeout value in seconds that the camera will wait for network settings from a DHCP server. The factory setting for `dhcp_timeout` is 15 seconds.

dns_domain_name

Type

string

Description

DNS domain name used by the camera.

Note

If this property is a null string, then the fully qualified domain name (`current_fqdn`) is determined dynamically by looking up the camera name in the DNS server (`nameserver`). This property can be set by the DHCP server.

Copyright © 2009

edge_enhancement

Type

string

Description

Sets the hardware built-in edge enhancement (image sharpening). The valid values are `default`, `alternate` and `off`.

Note

Not supported by Le045 and Intelligent (Li series) cameras.

ethaddr

Type

String, read-only

Description

Provides the factory set MAC address.

exposure

Type

double

Description

Accesses the exposure time in milliseconds used by the image sensor.

Notes

If `autoexposure=0`, the exposure time is fixed at the value set by this property.

If `autoexposure=1`, the exposure may vary in order to bring the `luminance` closer to the `autoexposure_target` value.

Not supported by Le045 and Intelligent (Li series) cameras.

Copyright © 2009

exposure_bias

Type

int

Description

This parameter is the offset to the automatic system gain using a value between -18 and 18.

Note

Applies only to the Le045 and Intelligent (Li series) cameras.

focus

Type

int

Description

Accesses the lens focal position.

Note

This property applies only to camera models that use a Canon 35 mm lens.

focus_max

Type

int

Description

Provides the maximum value of `focus` property.

Note

This property applies only to camera models that use a Canon 35 mm lens.

focus_min

Type

int

Copyright © 2009

Description

Provides the minimum value of `focus` property.

Note

This property applies only to camera models that use a Canon 35 mm lens.

frame_buffer_enable

Type

boolean

Description

Enable or disable analog video output.

Note

Applies only to the Le045 and Intelligent (Li series) cameras.

framerate

Type

double, read-only

Description

Provides the frame rate of the camera in units of frames per second.

Notes

This property may be affected by the image `width`, `height`, `exposure`, and `maximum_exposure`.

Network bandwidth may also affect perceived camera frame rate independent of this property.

gain

Type

double

Description

Accesses the camera's gain property.

Notes

Increasing the gain increases the `luminance`, which can help in low light conditions; however, a higher gain also substantially increases the noise in the image. Image compression is greatly and negatively affected by noisy images. It is recommended that this property be kept as low as possible.

If `autogain=0`, the gain value is fixed at the value set by this property.

If `autogain=1`, the gain value may vary in order to bring the `luminance` closer to the `autoexposure_target` value.

gain_blue

Type

double

Description

Provides manual control of the blue channel gain. This property is used to color/white balance the image and is adjusted when the `whitebalance` property is set.

Note

Not supported by Le045 and Intelligent (Li series) cameras.

gain_blue_max

Type

double

Description

Provides the maximum `gain_blue` value. This property can be set directly or by using the `whitebalance_preset` property.

Note

Not supported by Le045 and Intelligent (Li series) cameras.

Copyright © 2009

gain_blue_min

Type

double

Description

Provides the minimum `gain_blue` value. This property can be set directly or by using the `whitebalance_preset` property.

Note

Not supported by Le045 and Intelligent (Li series) cameras.

gain_green

Type

double

Description

Provides manual control of the green channel gain. This property is used to color/white balance the image and is adjusted when the `whitebalance` property is set.

Note

Not supported by Le045 and Intelligent (Li series) cameras.

gain_green_max

Type

double

Description

Provides the maximum `gain_green` value. This property can be set directly or by using the `whitebalance_preset` property.

Note

Not supported by Le045 and Intelligent (Li series) cameras.

Copyright © 2009

gain_green_min

Type

double

Description

Provides the minimum `gain_green` value. This property can be set directly or by using the `whitebalance_preset` property.

Note

Not supported by Le045 and Intelligent (Li series) cameras.

gain_red

Type

double

Description

Provides manual control of the red channel gain. This property is used to color/white balance the image and is adjusted when the `whitebalance` property is set.

Note

Not supported by Le045 and Intelligent (Li series) cameras.

gain_red_max

Type

double

Description

Provides the maximum `gain_red` value. This property can be set directly or by using the `whitebalance_preset` property.

Note

Not supported by Le045 and Intelligent (Li series) cameras.

Lumenera corporation
Copyright © 2009

gain_red_min

Type

double

Description

Provides the maximum `gain_red` value. This property can be set directly or by using the `whitebalance_preset` property.

Note

Not supported by Le045 and Intelligent (Li series) cameras.

gamma

Type

double

Description

Accesses the gamma property which affects the mid-tone brightness in the images.

Notes

Higher values of gamma give the image an overall brighter cast.

The value range for this property is from 0.01 to 100. The practical range is from 1 to 2. The typical value is around 1.4

gateway

Type

string

Description

Provides the gateway the camera will use on the next reboot when DHCP is disabled.

h264_enable

Type

boolean

Description

Enable or disable H.264 encoding for images.

Notes

H.264 encoding requires the following settings: `jpeg_enable=0`, `analytics_enable=0`, and `rtp_enable=1`.

Applies only to the Le045 and Intelligent (Li series) cameras.

height

Type

int

Description

Accesses the image height value in pixels.

Notes

This value must be a multiple of 8 with a minimum value of 16. If the specified height is not a multiple of 8, the camera will round down to the next lowest multiple of 8.

Immediately after a change in the image geometry (`x`, `y`, `width`, and/or `height`), wait at least 2*(1/`framerate`) or a subsequent `image` call may fail with an error.

image_stabilizer

Type

boolean

Description

Activate automatic image stabilization of the lens. Default is false (disabled). Set to true to power an IS (image stabilized) lens.

Copyright © 2009

Notes

Applies only to camera models that support Canon EF mount 35-mm lenses, such as the Le259 and Le11059.

Requires an IS lens with Canon EF mount.

imager_mono

Type

boolean, read-only

Description

Provides the image sensor type used by the camera. True if the sensor in the camera is a monochrome (black and white) imager. False if the imager is color.

intensity

Type

double

Description

Accesses the camera's numerical intensity value.

Notes

This property is similar to `contrast` except that the slope is centered at black instead of gray. Where `contrast` makes dark pixels darker and bright pixels brighter, this property brightens every pixel.

io_in_alarm_enable

Type

boolean

Description

Enables or disables detection of GPIO alarms.

Notes

When set, an alarm event may occur due to a signal applied to the GPIO hardware input to the camera. Detection of an actual alarm event is affected by `io_in_debounce_time`, `io_in_trigger_edge`, and `io_in_trigger_neg`.

io_in_debounce_time

Type

int

Description

GPIO input debounce time value in units of milliseconds.

Note

`io_in_debounce_time` is not used on the Li series cameras or the Le045.

io_in_trigger_edge

Type

boolean

Description

Reads and sets the trigger mode the GPIO input. It can be set to trigger on a rising/falling edge or a voltage level.

io_in_trigger_neg

Type

boolean

Description

When true, the GPIO input will trigger an alarm event based on a negative level or falling edge. When false, the alarm will be triggered on a positive level or rising edge.

ipaddr

Type

string

Description

Provides the IP address the camera will be used on the next reboot if DHCP is disabled.

Copyright © 2009

iris_type

Type

string

Description

Describes the type of iris that is installed on the camera. Values for this property are `none`, `dc`, and `calibrated`.

Notes

This property applies only to camera models that use a Canon 35 mm lens.

When `iris_type` is set to `calibrated`, the `aperture` property value is limited by `aperture_min` and `aperture_max`. If the lens is not properly installed, `aperture` will return the value -1.

When `iris_type` is set to `none` or `dc`, `aperture` will return the value -1.

ir_shuttle_position

Type

int

Description

Provides the IR filter shuttle position. A value of 1 indicates night mode, where the IR-cut filter is removed from the light path.

is_running

Type

boolean, read-only

Description

Provides the current status of the JPEG compression process. A true value states that the encoder is busy converting images into JPEGs.

jpeg_buffer_size

Type

int, read-only

Description

Provides the memory in bytes allocated to each JPEG image in the rolling buffer. This value is inversely proportional to the `number_of_buffers` value.

jpeg_quality

Type

double

Description

Specifies the JPEG quality level using a value between 0.2 and 99.

Note

The larger the number provided for this property the larger and higher quality the resulting JPEG image files produced by the camera. This property becomes read-only when `auto_jpeg_quality=1`.

led

Type

boolean

Description

Controls the state of the orange LED on the network connector. When true, the LED is active.

lighting_frequency

Type

double

Description

AC mains frequency for artificial lighting in Hz.

Note

This property limits the exposure to a multiple of a half cycle time (8.33ms for 60Hz, 10.0ms for 50Hz), in order to eliminate dark and light bands from appearing in the image. It should be configured when the camera is placed in an environment where artificial AC based light is used to illuminate the scene.

Copyright © 2009

For countries using 60 Hz electrical power such as North America, this property should be set to 60.

For 50Hz electrical power such as Europe, set it to 50.

For all DC light sources such as sunlight, set this property to 0.

link_local_ip

Type

string

Description

Provides the link-local IP address that will be used when `use_link_local=1`.

low_light_criteria

Type

string

Description

Describes a list of criteria that describes the night mode threshold needed to set the camera into its night mode.

Notes

A typical value is “gain > 32”.

The expression is phrased using comparison operators =,<, >, <=, >=, != (does not equal), & (AND), | (OR) and arithmetic symbols +, -, *, /, %. The expression elements must be separated by at least 1 space.

The camera evaluates this expression once per second to determine if the night mode threshold has been achieved. The criteria must be valid for a time in seconds given by `low_light_stable_duration` in order for the camera to switch to night mode by applying the `low_light_settings`.

The `exposure`, `low_light_exposure_threshold`, `gain`, `low_light_gain_threshold`, `luminance`, and `low_light_luminance_threshold` values are available for comparison by name in the provided expression.

low_light_settings

Type

string

Description

Describes a set of properties and values that are applied when the camera switches to night mode.

Note

The settings are provided as a semicolon separated list. The camera uses `low_light_criteria` and `low_light_stable_duration` to determine when to switch to night mode.

low_light_stable_duration

Type

double

Description

Defines a time in seconds for which `low_light_criteria` must be valid before `low_light_settings` are applied.

luminance

Type

double, read-only

Description

Provides the current luminance (image intensity) value expressed as a percentage of full bright image.

Note

This property is affected by the settings for `auto_x`, `auto_y`, `auto_width`, `auto_height`, `auto_algorithm`, `autoexposure`, `autogain`, and `autoiris`.

Copyright © 2009

MAC

Type

string, read-only

Description

Provides the camera's Ethernet MAC address.

max_height

Type

int, read-only

Description

Provides the maximum height supported by the imager in the camera.

max_jpeg_size

Type

int

Description

Defines the maximum JPEG image size in bytes that should be used by the camera when compressing the video images.

Notes

The maximum JPEG size that is used by the auto_jpeg_quality algorithm is the minimum value of the following:

`max_jpeg_size`, or

`projected_network_bandwidth` divided by the minimum value of either `projected_network_framerate` or `framerate`.

When set to 0, the `auto_jpeg_quality` algorithm will use the `jpeg_buffer_size` value instead.

For the Le045 and the Intelligent (Li series) cameras, the `projected_network_bandwidth` and `projected_network_framerate` properties are not supported. See `target_jpeg_size`.

max_subsample

Type

int, read-only

Description

Provides the maximum subsampling factor supported by the imager used in the camera.

max_width

Type

int, read-only

Description

Provides the maximum width supported by the imager used in the camera.

maximum_exposure

Type

double

Description

Defines the limit of the exposure range when `autoexposure=1`.

Note

Setting this property to 0 sets the exposure limit so that the frame rate is not affected.

maximum_framerate

Type

double

Description

Defines a limit on the camera's maximum frame rate.

Copyright © 2009

Note

To disable the limit set this property to 0 to disable.

maximum_gain

Type

double

Description

Defines a limit to the gain range when `autogain=1`.

maximum_gain_limit

Type

double, read-only

Description

The absolute maximum gain supported by the camera model in use.

median_filter

Type

boolean

Description

Enables or disables the median filter correction. Default is false.

Notes

The median filter is used to correct for over-reactive pixels in an image typically arise due high gain situations when working in low light levels.

The filter compares each pixel to its neighbors to determine if it is representative of its surroundings. If not, the filter will replace the value of the pixel by the mean of the neighboring pixels.

Not supported by Le045 and Intelligent (Li series) cameras.

Copyright © 2009

mem_free

Type

int, read-only

Description

Provides the amount of free RAM memory in kilobytes available to the operating system.

min_jpeg_size

Type

int

Description

Defines the minimum desired JPEG image size in bytes for use with `auto_jpeg_quality`.

minimum_exposure

Type

double

Description

Limits the exposure value when `autoiris=1`.

Notes

If `autoexposure=0`, this value is ignored.

If `autoiris=1` and `autoexposure=1` then this value is used as the lower limit for the exposure.

motion_alarm_area

Type

double

Copyright © 2009

Description

Defines the percentage of the pixels in the motion detection area that must change before a motion alarm event occurs.

Note

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_alarm_enable

Type

boolean

Description

Enable and disable motion detection alarms.

Notes

Motion is defined as a minimum number of pixels changing defined by the `motion_alarm_area` property by a minimum amount defined by the `motion_sensitivity` property in a specified area defined by the `motion_map`.

The area of interest is a grid of rectangular areas. Each area can be independently enabled or disabled. Only enabled areas will be tested for motion.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_email_recipient

Type

string

Description

Provides the recipient email address for motion alarm events when the default `motion_email_script` is in use.

Note

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_email_script

Type

string

Description

Defines a script that is executed when a motion alarm event occurs.

Notes

The factory-set default value of this script uses the additional properties `motion_email_subject`, `motion_email_server`, `motion_email_recipient`, and `alarm_delay` to create and send an email message when motion is detected. This script can be modified to change the body of the message, take additional actions and/or include variable properties in the subject line.

The contents of the script are as follows:

```
#!/bin/sh
mail -s "`l get motion_email_subject`"
-f motion_alarm@`l get current_fqdn`
-S `l get motion_email_server`
-H `l get current_fqdn` "`l get motion_email_recipient`"
<<EOF
Motion detected.  Centroid is `l get motion_x`,`l get
motion_y`.  Area is `l get motion_area`%.
EOF
```

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_email_server

Type

string

Description

Defines an email server that is used to send an email message upon a motion alarm event when the default `motion_email_script` is in use.

Note

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

Copyright © 2009

motion_email_subject

Type

string

Description

Defines the subject of the email message that is sent upon a motion alarm event when the default `motion_email_script` is in use.

Note

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_gap

Type

double

Description

Defines the delay between frames being evaluated for motion detection. This property is measured in milliseconds with a value between 0 and 255 frames.

Notes

This property is similar to `motion_skip`.

If this value is set too low, objects that are moving slowly may not get detected.

If it is set too high, fast objects that can move through the region of interest during the skipped frames and may not get detected correctly.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_map

Type

string

Description

Enables and/or disables the rectangles in the motion map.

Notes

The motion map is a grid of `motion_map_width` by `motion_map_height` that is superimposed on top of the current viewing window (or the entire image sensor if `motion_map_windowed` is true). Each grid rectangle is either enabled or disabled by this property.

The map is specified as a string of hexadecimal digits, each bit corresponding to a single grid location as 1 (active) or 0 (inactive). The maximum value of the `motion_map` is 2^(`motion_map_width` * `motion_map_height`)-1, for example 11111111 binary, FFFF hexadecimal, or 255 decimal for a 4 x 4 grid.

The most significant bit of this value corresponds to the upper left motion map grid element. The following bits correspond to the next grid element to the right until the end of the row. The following bits continue on the next row to the last bit which defines the bottom right grid element.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_map_height

Type

double

Description

Defines the number of horizontal rows in the `motion_map`.

Notes

This property defines the number of rows in the motion map. It does not refer to the height of each rectangle in the motion map or the height of the entire motion map.

The height of each rectangle in the motion map if `motion_map_windowed=0` is `height` / `motion_map_height` rounded to the nearest multiple of 4.

If `motion_map_windowed=1` then the height is `max_height` / (`subsampled` + 1) / `motion_map_height` rounded to the nearest multiple of 4.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_map_width

Type

double

Copyright © 2009

Description

Defines the number of vertical columns in the `motion_map`.

Notes

This property defines the number of columns in the motion map. It does not refer to the width of each rectangle in the motion map or the width of the entire motion map.

The width of each rectangle in the motion map if the `motion_map_windowed=0` is `width` / `motion_map_width` rounded to the nearest multiple of 4.

If `motion_map_windowed=1` then the height is `max_width` / (`subsampled` + 1) / `motion_map_width` rounded to the nearest multiple of 4.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_map_windowed

Type

Boolean

Description

Defines the motion map coverage over the camera's sensor.

Notes

When this property is set, only the viewable area of the camera's sensor is used for motion detection. This means that if the camera is currently sub-windowed down to only view one quarter of the full image dimensions, only this sub-windowed section is used for motion detection.

When this property is cleared, the camera uses the entire viewing area of the sensor for motion detection.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_sensitivity

Type

double

Description

Defines the sensitivity of motion detection. Values from 0 to 100 are valid, with 100 being the most sensitive, meaning that the pixel has to change by the least

amount to be counted as a changed pixel. Not supported by Le045, Le11059 and Intelligent (Li series) cameras.

motion_skip

Type

int

Description

Defines the delay between frames being evaluated for motion detection. This value may be between 0 and 255, and is measured in frames.

Notes

This property is similar to `motion_gap`.

If this value is set too low, objects that are moving slowly may not get detected.

If it is set too high, fast objects that can move through the region of interest during the skipped frames and may not get detected correctly.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

motion_x

Type

int, read-only

Description

Defines the x location of the centroid of motion in the detection window during the last alarm.

Notes

The centroid can be located in a non-enabled area. For example, if the only enabled rectangles are the four corners of the imager and each corner registers motion at the same time, the centroid of the motion would be near the center of the imager which would not be inside of any of the four corner's rectangular areas.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

Copyright © 2009

motion_y

Type

int, read-only

Description

Defines the y location of the centroid of motion in the detection window during the last alarm.

Notes

The centroid can be located in a non-enabled area. For example, if the only enabled rectangles are the four corners of the imager and each corner registers motion at the same time, the centroid of the motion would be near the center of the imager which would not be inside of any of the four corners' rectangular areas.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

name

Type

string

Description

Provides the camera's hostname.

Note

If the DHCP server passes a hostname, this hostname overrides the value in this property.

nameserver

Type

string

Description

Provides the name server that the camera will use after the next reboot if DHCP is disabled (`use_dhcp=0`).

netmask

Type

string

Description

The subnet mask the camera will have after the next reboot if DHCP is disabled (`use_dhcp=0`).

normal_light_criteria

Type

string

Description

Describes a list of criteria that describes the day mode threshold needed to set the camera into its day mode.

Notes

A typical value is “gain < 16”.

The expression is phrased using comparison operators =,<, >, <=, >=, != (does not equal), & (AND), | (OR) and arithmetic symbols +, -, *, /, %. The expression elements must be separated by at least 1 space.

The camera evaluates this expression once per second to determine if the night mode threshold has been achieved. The criteria must be valid for a time in seconds given by `normal_light_stable_duration` in order for the camera to switch to night mode by applying the `normal_light_settings`.

The `exposure`, `normal_light_exposure_threshold`, `gain`, `normal_light_gain_threshold`, `luminance`, and `normal_light_luminance_threshold` values are available for comparison by name in the provided expression.

normal_light_settings

Type

string

Copyright © 2009

Description

Describes a set of properties and values that are applied when the camera switches to day mode.

Note

The settings are provided as a semicolon separated list. The camera uses `normal_light_criteria` and `normal_light_stable_duration` to determine when to switch to day mode.

normal_light_stable_duration

Type

double

Description

Defines the time in seconds for which `normal_light_criteria` must be true before `normal_light_settings` will be applied.

ntp_servers

Type

string

Description

Describes a list of NTP (Network Time Protocol) servers as a whitespace separated list.

Note

Set to a null string to disable NTP. When enabled, it is recommended to enable `clock_synchronize_from_kernel_to_rtc` as well.

number_of_buffers

Type

int

Description

Defines the number of JPEG images stored in the rolling buffer memory.

Notes

The `jpeg_buffer_size` in bytes is inversely proportional to `number_of_buffers`.

To continue processing new images when `number_of_buffers` is increased, it may be necessary to reduce `jpeg_quality` to keep the image size less than `jpeg_buffer_size`.

Changes to this property only take effect if the `save_settings` command is first used to store the change to non-volatile memory and then the `reset` command is used to reboot the camera.

number_of_buffers_raw

Type

int

Description

Defines the number of raw (uncompressed) images the rolling buffer memory can hold.

Note

This property must be set to a number larger than 2 for the `image_raw` or `image_yuv` functions to take effect. Changes to this property only take effect on reboot of the camera.

output_select

Type

string

Description

Defines the type of output available on the camera's general purpose output pins. The valid settings are:

`on`: Current can flow through the output connection.

`off`: Output transistor is turned off.

`alarm`: Output a pulse when an alarm occurs. The pulse duration is defined by the `strobe_length` property. The alarm may be either a video motion detection alarm or a signal on the general purpose input pins.

Copyright © 2009

`start_of_readout`: Output a pulse once per frame, when the image sensor starts transferring data to the hardware CODEC, that is, after line 1 of the image completes its exposure. This is the appropriate setting to use to trigger external strobe lighting with cameras using rolling shutter sensors. The limitation is that line 1 of the image will never be exposed to the flash. Not supported by Le045 and Intelligent (Li series) cameras.

`strobe`: Output a pulse once per frame, when the image sensor signals that all pixels are being exposed. This is the appropriate setting to use to trigger external strobe lighting with cameras using global shutter sensors. Not supported by Le045 and Intelligent (Li series) cameras.

`watchdog_ping`: Output an alternating a series of on/off pulses when a ping of the IP address specified in the `watchdog_ping` property fails (times out). This provides an alarm in the event of network communications failure.

privacy_map

Type

string

Description

Specifies rectangular portions of the image that will be hidden from view in all the images produced by the camera.

Notes

The privacy map is a grid of `privacy_map_width` by `privacy_map_height` that is superimposed on top of the current viewing window (or the entire image sensor if `privacy_map_windowed` is true). Each grid rectangle is either enabled or disabled by this property.

The map is specified as a string of hexadecimal digits, each bit corresponding to a single grid location as 1 (active) or 0 (inactive). The maximum value of the `privacy_map` is 2^(`privacy_map_width` * `privacy_map_height`)-1, for example, 11111111 binary, FFFF hexadecimal, or 255 decimal for a 4 x 4 grid.

The most significant bit of this value corresponds to the upper left privacy map grid element. The following bits correspond to the next grid element to the right until the end of the row. The following bits continue on the next row to the last bit which defines the bottom right grid element.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

privacy_map_enable

Type

boolean

Description

Enables or disables the entire `privacy_map`.

Note

Not supported by Le11059 cameras.

privacy_map_height

Type

double

Description

Defines the number of horizontal rows in the `privacy_map`.

Notes

This property defines the number of rows in the privacy map. It does not refer to the height of each rectangle in the privacy map or the height of the entire privacy map.

The height of each rectangle in the privacy map if the `privacy_map_windowed=0` is `height` / `privacy_map_height` rounded to the nearest multiple of 4.

If `privacy_map_windowed=1` then the height is `max_height` / (`subsample` + 1) / `privacy_map_height` rounded to the nearest multiple of 4.

Not supported by Le11059 cameras.

privacy_map_width

Type

double

Description

Defines the number of vertical columns in the `privacy_map`.

Copyright © 2009

Notes

This property defines the number of columns in the privacy map. It does not refer to the width of each rectangle in the privacy map or the width of the entire privacy map.

The width of each rectangle in the privacy map if the `privacy_map_windowed=0` is `width` / `privacy_map_width` rounded to the nearest multiple of 4.

If `privacy_map_windowed=1` then the height is `max_width` / (`subsample` + 1) / `privacy_map_width` rounded to the nearest multiple of 4.

Not supported by Le11059 cameras.

projected_network_bandwidth

Type

int

Description

Defines the estimated network bandwidth available in units of Bytes per second.

Notes

The target JPEG image file size in bytes is determined by this property along with the `projected_network_framerate` property.

Does not apply to the Le045 or the Li series cameras. Use the `target_jpeg_size` property instead.

projected_network_framerate

Type

int

Description

Provides the anticipated frame rate in frames per second to be transmitted over the network. This is different from the `framerate` property where this value defines the rate at which the camera captures new images.

Notes

The target JPEG image file size in bytes is determined by this property along with the `projected_network_bandwidth` property.

Does not apply to the Le045 or the Li series cameras. Use the `target_jpeg_size` property instead.

ptz_enable

Type

boolean

Description

Enables or disables PTZ (pan-tilt-zoom) functions.

Notes

Applies only to Pan Tilt Zoom (PTZ) systems that use the camera's RS232 serial port to interface the network camera to a pan-tilt motor and a motorized zoom lens. Calling `ptz_command` or `ptz_preset` sends a command to a PTZ controller board via the camera's RS232 serial port.

Set `ptz_enable` to be true, `ptz_protocol`=pelco_d, `serial_use`=accessory, and `serial_speed`=2400 to work with a Lumenera factory-ready PTZ system.

Resetting the camera to factory defaults will disable PTZ functions.

ptz_protocol

Type

string

Description

Specifies the PTZ control protocol as: pelco_d, pelco_p, or vt.

Note

The normal Lumenera factory default setting is `ptz_protocol`=pelco_d.

Set `ptz_enable` to be true, `ptz_protocol`=pelco_d, `serial_use`=accessory, and `serial_speed`=2400 to work with a Lumenera factory-ready PTZ system.

Applies only to Pan Tilt Zoom (PTZ) systems that use the camera's RS232 serial port to interface the network camera to a pan-tilt motor and a motorized zoom lens. Calling `ptz_command` or `ptz_preset` sends a command to a PTZ controller board via the camera's RS232 serial port.

Copyright © 2009

ram_size

Type

int, read-only

Description

Provides the amount of SDRAM that has been initialized in the camera. Units are bytes.

raw_mode

Type

boolean

Description

Disabled the image processing function and places the camera in RAW image mode.

Note

When in RAW mode, the JPEG hardware CODEC is turned off, and the RAW imager data may be accessed using the `image.raw` or `image_yuv` commands.

rtp_enable

Type

boolean

Description

Enables or disables the RTP daemon.

Note

RTP streaming does not necessarily end when clients disconnect, so explicitly disabling RTP may be required to reclaim the bandwidth. Once enabled, RTSP clients may connect via `rtsp://camera/` or `http://camera/rtp.sdp`. For model Le045 and Intelligent (Li series) cameras, additional video stream connections include `rtsp://camera/mjpeg` and `rtsp://camera/h264`.

The maximum `width` supported for RTP streaming is 2048 pixels. RTP streaming of images with wider dimensions will produce incorrectly formatted images.

Copyright © 2009

rtp_framerate

Type

double

Description

Defines the frame rate used for RTP transfer.

Note

Network bandwidth may affect the perceived frame rate.

rtp_packet_size

Type

int

Description

The size of RTP packets in units of bytes.

Note

A size of 1450 ensures that Ethernet fragmentation does not occur. A size of 65400 provides much faster transfers. It is recommended to use a packet size of 1450 to minimize the fragmentation at a lower level in the protocol stack.

rtp_port

Type

int

Description

Defines the RTP base port number to use for transmissions, which must always be an even number.

Note

This value should not be confused with the RTSP port number, which is always at the IANA standard location of 554.

Copyright © 2009

saturation

Type

double

Description

Provides access to control the boldness of color in the images.

Notes

This property can take values between -100 and +100. The most common settings for this value are 0 (normal color) and -100 (monochrome).

Not supported by Le045 and Intelligent (Li series) cameras.

saturation_bias

Type

double

Description

Adjusts the amount of color in the image. This parameter takes a value between -100 and +100.

Notes

Applying a negative value will decrease the amount of color and cause the video to change towards black and white. A positive value will increase the amount of color in the video.

Applies only to the Le045 and Intelligent (Li series) cameras.

serial_echo

Type

boolean

Description

Enables or disables local echo on the camera's serial port.

serial_flow

Type

string

Description

Defines the flow control method for the serial port. The available options are `none`, `sw`, `hw`, and `both`.

serial_speed

Type

int

Description

Defines the speed of the serial port.

serial_use

Type

string

Description

Enables or disables the serial port for console or for an accessory. The valid values are `console`, `accessory`, or `none`.

Notes

To apply changes first call `save_settings` to store the change to persistent memory and then reboot the camera.

In the accessory mode, the camera serial port stops sending characters as soon as the TCP connection closes, even if some characters have been received but have not been echoed on the serial port.

show_motion_detected

Type

boolean

Copyright © 2009

Description

Displays the motion that was detected by the camera by applying a white overlay that appears as a “ghosted image”.

Notes

Since images sent by all methods will display the detected motion, this mode is intended to be used only as a setup and calibration tool. Both `show_motion_windows` and `show_motion_detected` cannot be set simultaneously.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

show_motion_windows

Type

boolean

Description

Displays the selected elements of the motion map grid that are defined in the motion map set in the camera. The regions are highlighted with a grey rectangle.

Notes

Since images sent by all methods will display the detected motion, this mode is intended to be used only as a setup and calibration tool. Both `show_motion_windows` and `show_motion_detected` cannot be set simultaneously.

Not supported by Le045, Intelligent (Li series), or Le11059 cameras.

strobe_length

Type

double

Description

Defines the amount of time in milliseconds that the output is on when the `strobe` function is called or when an alarm strobe is fired. This property is ignored when `output_select=strobe`.

subsampled

Type

int

Description

Specifies the number of pixels that should be skipped (decimation or subsampling) or combined (binning) for each pixel that is displayed.

Note

A change in `subsampled` involves a change in the image geometry/dimensions. Immediately after a change in the image geometry (`x`, `y`, `width`, and/or `height`), wait at least 2*(1/`framerate`) or a subsequent `image` call may fail with an error.

target_jpeg_size

Type

int

Description

The target JPEG image file size in bytes that is used by the `auto_jpeg_quality` algorithm. If set to the special value of zero, `target_jpeg_size` will be set to 90% of `jpeg_buffer_size`.

Note

Applies only to the Intelligent (Li series) cameras and the Le045.

telnetd

Type

boolean

Description

When `telnetd` is true, the telnet daemon is enabled. When `telnetd` is false, the telnet daemon is disabled.

Copyright © 2009

temperature

Type

double, read-only

Description

Provides the approximate internal temperature of the camera measured on the image sensor circuit board. Units are degrees Celsius (ºC).

timezone

Type

string

Description

Provides the camera's the time zone, in standard Unix (SUSv2) format.

Notes

See http://www.opengroup.org/onlinepubs/007904975/basedefs/xbd_chap08.html for the precise definition of the possible values.

Some examples are:
JST-9 means that Japanese Standard Time (JST) is 9 hours later than Coordinated Universal Time (UTC) and does not have daylight saving time.

MST7MDT means that Mountain Standard Time (MST) is 7 hours earlier than UTC. Both standard time and daylight saving time apply to this locale. By default Mountain Daylight Time (MDT) is one hour ahead of MST.

NST3:30NDT1:30 means that Newfoundland Standard Time (NST) is 3.5 hours earlier than Coordinated Universal Time (UTC). Both standard time and daylight saving time apply to this locale. Newfoundland Daylight Time is 1.5 hours earlier than Coordinated Universal Time (UTC).

use_dhcp

Type

boolean

Description

Enables or disables DHCP support.

Note

When `use_dhcp` is true, the camera attempts to obtain its network settings from a DHCP server on its next reboot.

When `use_dhcp` is false, the camera can be set to a static IP address as follows.

- Set IP address, netmask, gateway (optional), nameserver (optional).
- Disable DHCP.
- Save Settings to the flash memory.
- Reboot the camera.

Example CGI commands are as follows. Property values are URL escaped.

```
/cgi-bin/set?ipaddr=123%2E123%2E123%2E123
/cgi-bin/set?netmask=255%2E255%2E255%2E0
/cgi-bin/set?gateway=123%2E123%2E123%2E1
/cgi-bin/set?nameserver=123%2E123%2E123%2E5
/cgi-bin/set?use_dhcp=0
/cgi-bin/save_settings
/cgi-bin/reset
```

The camera will reboot with the new network settings applied.

use_link_local

Type

string

Description

Enables or disables the link local address.

Note

When set, the camera is configured with a link local address on its next reboot. Setting this property to 0 prevents the assignment of link local address.

user_password

Type

string

Copyright © 2009

Description

Access to the user password for the camera. This property can be used to set the password.

Notes

If this property is not empty, a password is needed to access any of the camera configuration commands in the /cgi-usr/ directory. To remove a password, set it to the null (empty) string. If the password starts with "1", the string is assumed to be the MD5 hash of the password rather than the password itself. The expected username is `user`. The `user` username and password may also be used to access the user sections of the camera.

Getting this property will return either `1` if a password is set or `null`.

video_mode

Type

string

Description

Specifies the video format of the composite video output as `ntsc` (default) or `pal`.

Notes

Applies only to the Intelligent (Li series) cameras and the Le045.

The `video_mode` property determines the boot configuration of the camera and is normally set at the factory.

Use the `save_settings` command to store any change to `video_mode` in persistent memory. Use the `reset` command to reboot the camera in the new `video_mode`.

watchdog_codec

Type

boolean

Description

Enables or disables the check for hardware CODEC and imager.

Notes

If set, the camera will check the hardware CODEC and imager status before kicking the watchdog. If the watchdog timer is exceeded because the watchdog is not reset, the camera will reboot.

Enabling this property is not recommended under normal conditions because, if the imager fails, the camera will constantly reboot making communication with the camera very difficult. An imager failure is generally not something a reboot will fix.

watchdog_enable

Type

boolean

Description

Enables or disables the watchdog.

Note

The watchdog will periodically check various system parameters. If these parameters indicate that the camera is functioning as expected, the timer is reset. If the timer expires before it is reset, the camera will reboot.

watchdog_http

Type

boolean

Description

Enables or disables the check to ensure that the web server is still responsive.

Note

If set, the camera will check the web server's status before kicking the watchdog. If the watchdog timer is exceeded because the watchdog is not reset, the camera will reboot.

watchdog_ping

Type

string

Copyright © 2009

Description

An IP address to ping in order to automatically test network communications.

Notes

The camera will ping any IP address specified. A correct reply to the ping is required in order to reset the watchdog timer. If the ping consistently fails, the watchdog timer will exceed its limit and the camera will reboot.

Using this property is not recommended under normal conditions. Setting this property is only useful if the camera is difficult to access and resetting the camera is likely to fix network connectivity issues.

whitebalance

Type

string

Description

Provides access to the white balance algorithm in use. Possible values for this property include `single`, `continuous`, and `off`.

Note

When this property is set to `off`, current balance is maintained.

A single, one shot white balance is performed when this property is set to `single`.

An ongoing repetitive white balance is enabled when set to `continuous`.

whitebalance_preset

Type

string

Description

Defines a specific white balance preset for a given lighting environment. Possible values include `default`, `wide`, `fluorescent`, `daylight`, and `user`.

Note

Each preset limits the color gain ranges by specifying preset values of the properties `gain_red_min`, `gain_red_max`, `gain_green_min`, `gain_green_max`, `gain_blue_min`, and `gain_blue_max`.

When this property is set to `user`, a custom set of color gain limits can be specified.

width

Type

int

Description

Defines the width of the image in pixels.

Notes

This property must be a multiple of 32 and will round down any value provided to the next lowest valid multiple of 32.

The maximum `width` supported for RTP streaming is 2048 pixels. RTP streaming of images with wider dimensions will produce incorrectly formatted images.

Immediately after a change in the image geometry (`x`, `y`, `width`, and/or `height`), wait at least 2*(1/`framerate`) or a subsequent `image` call may fail with an error.

x

Type

int

Description

Defines the horizontal offset of the image in pixels from the left edge of the camera's sensor.

Notes

The value provided must be an even number and will round down any value provided to the next lowest valid even number.

Immediately after a change in the image geometry (`x`, `y`, `width`, and/or `height`), wait at least 2*(1/`framerate`) seconds or a subsequent `image` call may fail with an error.

Copyright © 2009

x_mirror

Type

boolean

Description

Provides access to mirror the image, flipping the image from left to right.

y

Type

int

Description

Defines the vertical offset of the image in pixels from the top edge of the camera's sensor.

Notes

The value of y must be a multiple of 8 pixels. Any value specified will be rounded down to the next lowest multiple of 8.

Immediately after a change in the image geometry (`x`, `y`, `width`, and/or `height`), wait at least 2*(1/`framerate`) seconds or a subsequent `image` call may fail with an error.

y_mirror

Type

boolean

Description

Provides access to flip the image, flipping the image from bottom to top.

A1

Appendix: Interpreting EXIF headers

A1.1 General information on EXIF header format

Using v.1.6 or higher firmware, the header for each JPEG image generated by the Lumenera Network cameras is fully compliant with the EXIF standard. The header includes time stamp information to microsecond precision.

The formatting is as specified in the EXIF 2.2 standard:

official http://www.jeita.or.jp

unofficial http://www.exif.org/

The human-readable time in the `DateTime` and `SubSecTime` fields is specified for the time zone configured on the camera using the `timezone` property of the API.

The time in the `MakerNote` is recorded as an array of 2 binary long integers. The elements are the seconds and the microseconds in the UNIX epoch time, tagged as field number `0xFDE8`.

Utilities for reading EXIF header information are available from:

http://home.cfl.rr.com/genecash/digital_camera.html

http://www.exifer.friedemann.info/

A1.2 Detailed example of EXIF header

Below is an example of the EXIF-compliant header and how it can be parsed to retrieve its contents.

The tag `0x927C` indicates `MakerNote`.

The field type for this tag is `0x7` meaning undefined format.

The length of the field is `0x26` or 38 bytes

Copyright © 2009

The value offset is `0x60` which places the start byte of the `MakerNote` at byte 108 of the header.

Below are the 38 bytes of `MakerNote` starting at byte 108.

Table 4-1 `MakerNote` Field Descriptions in the EXIF header

BYTE	VALUE	USE
1-4	00 02 FD E8	Identifying number.
5-6	00 04	Type of data = 4, unsigned 32 bit integer.
7-10	00 00 00 02	Number of data elements =2.
11-14	00 00 00 7E	Pointer to time value relative to the start of tagged data. See byte 31 below.
15-16	FD E9	Identifying number.
17-18	00 04	Type of data = 4, unsigned 32 bit integer.
19-22	00 00 00 01	Number of data elements = 1.
23-26	00 A4 2D 5C	Data element, image sequence number.
27-30	00 00 00 00	Not used.
31-34	43 E9 2C 3E	Time in seconds.
35-38	00 00 73 F0	Partial time in microseconds.

A2

Glossary

ABC Auto Brightness Control. As the image grows darker, ABC will apply the following measures:

1. Open the dc-auto iris lens, if present.
2. Increase exposure up to the `autoexposure_knee`.
3. Increase (analog) gain up to the `autogain_knee`.
4. Increase exposure up to the `maximum_exposure`.
5. Increase gain up to the `maximum_gain`.

As the image gets darker, the steps above are done in reverse order.

API Application Programming Interface

CGI Common Gateway Interface. Refer to Section 2.4.2.

CRLF or **=\r\n** Denotes the combination of the carriage return and line feed characters

DC iris Cameras in the CS-mount format support DC-iris (galvanometric) lenses. Video auto iris lenses are not supported.

DHCP Dynamic Host Control Protocol. By default, a camera will attempt to locate a DHCP server from which it will obtain an IP address.

DNS Domain Name Server

EXIF Exchangeable Image File Format. The Lumenera Network cameras produce JPEG images with an EXIF-compliant header. The metadata tags include information identifying Lumenera as the manufacturer, a frame number, the time in seconds, and the time in microseconds for each image. For details, refer to Appendix A1.

FQDN Fully Qualified Domain Name

FTP File Transfer Protocol. In the camera this may be used to transfer images to a server on demand, upon motion, or upon receiving a signal on the GPIO input.

Copyright © 2009

Gamma	A parameter that specifies a nonlinear characteristic for the mid-tone brightness level in the image. A value of 1.4 is typical. See `gamma` in the API property reference.
GPIO	General Purpose Input and Output. The cameras are equipped with one pair of terminals for input and another for output. Both are optically isolated to minimize the risk of accidental electrical damage to the camera.
HTTP	Hyper Text Transfer Protocol. The data throughput for Lumenera Network cameras over HTTP is 1.5 MB/s and about 50% greater over UDP.
IP	Internet Protocol.
IR	Infrared. The cameras use silicon-based imagers and as such are sensitive to near-infrared wavelengths from about 700 nm to about 1100 nm. For some color cameras, the '-DN' day/night version provides the ability to detect IR light. An infrared-cut-off filter is used in daylight conditions to provide accurate color rendition. An uncoated infrared-transparent glass is used during night time conditions to boost the sensitivity of the camera. Images are normally rendered as monochrome (black and white) in night mode.
JPEG	Joint Photographic Experts Group. Denotes an image compression format that is commonly used in computing. The compression procedure includes encoding the image in the YCbCr color space, downsampling color (chroma, Cb, Cr) information, dividing the image into an array of 8x8 pixel cells, computing the discrete Fourier cosine transform of each cell, quantization of brightness information, and entropy coding. The associated file format used in computing is called JFIF for JPEG File Interchange Format. See also `EXIF`.
MAC	Media Access Control address. The MAC consists of six bytes expressed as hexadecimal values. For Lumenera products, the MAC is of the form 00:0B:E2:10:04:AB. The first three octets (00:0B:E2) identify Lumenera as the manufacturer and the last three octets (10:04:AB in this example) vary from one camera to the next. The MAC can be used to derive the link-local IP address of the camera by converting the last two octets to decimal values using the following formula 169.254.(93+MAC5).MAC6. For the example shown, MAC5=4, MAC6=AB=171 (decimal), so the link-local address is169.254.97.171.
MD5	A commonly used 128-bit cryptographic hash function. User and Admin passwords on Lumenera cameras can be set to

	use the MD5 hash of the password. See `user_password` and `admin_password` for more information.
MJPEG	Motion JPEG. A video stream composed of a series of JPEG images, in which every frame updates the entire field. While other formats that update only the changes between key frames can consume less bandwidth under favorable conditions, MJPEG provides an optimal balance between data compression and image quality under all conditions and a complete update of every image.
NPH	No Process Header. Refer to section 2.4.2 on the Common Gateway Interface (CGI).
NTP	Network Time Protocol. The API property `ntp_servers` is provided to allow one to specify the IP addresses of one or more NTP servers from which the camera's real-time clock may be updated.
RAW	The term `RAW image` refers to the binary image data as recorded by the image sensor. No corrections or image processing are applied to this image. In particular, no JPEG compression has been applied. For color images, no demosaicing of the Bayer pattern of color filters on the pixels has been applied to generate RGB values for each pixel of the image. See `raw_mode` for more information.
RFC	Request For Comments. A numbered series of memoranda published on the Internet describing innovations in computer network engineering. The Internet Engineering Task Force (IETF) may adopt some RFCs as standards.
RGB	Red, Green and Blue. A color image in which each pixel is assigned Red, Green, and Blue components. In RGB color image capture, each pixel of the image sensor acquires only the signal for only one color channel. Color filters on each pixel are arranged in a regular mosaic pattern. Values for the two missing color values for each pixel are derived by interpolating between the values of neighboring pixels. Compare with **YUV**.
RTC	Real-Time Clock. The on-camera clock.
RTP / RTSP	Real-Time Transfer Protocol / Real Time Streaming Protocol.
Telnet	Teletype Network. A network protocol that provides means to communicate with a remote computer through a command-line console client application. The camera supports a telnet daemon for this purpose. See `telnetd` for more information.

Copyright © 2009

UDP	User Datagram Protocol. The data throughput for Lumenera Network cameras over HTTP is 1.5 MB/s and about 50% greater over UDP. The speed of UDP is due to the lack of overhead for checking if every packet reaches its destination. Packets may arrive out of order or fail to arrive without notice.
URI or **URL**	Uniform Resource Identifier, Uniform Resource Locator. A URL such as http://www.lumenera.com is a type of URI that identifies a resource and the particular representation of that information.
UTC time	Coordinated Universal Time. The Lumenera Network cameras use UTC time to account for the timezone and changes between Daylight and Standard Time.
YCbCr and **YUV**	Alternative color spaces to `RGB` in which the image components are the luminance or “luma” (Y) and the color (“chroma”) values Cb and Cr, or U and V, respectively. The Y component is related to the overall brightness and is a combination of the R, G, and B values. In an analog composite video signal, the Y component is sufficient to display monochrome (black & white) images.
YUY2	A format for image information in which luma (Y, “brightness”) is encoded for each pixel, but chroma (Cb, Cr, “color”) information is provided only for every second pixel.
XML-RPC	Extensible Markup Language-Remote Procedure Call. A simple protocol for making remote procedure calls. XML is used to encode the calls and HTTP is used for transport. The installation CD-ROM includes XMLRPC libraries for C++ and Visual Basic.

7 Capella Court, Ottawa, ON, Canada K2E 8A7

Phone: (613) 736-4077 | Fax: (613) 736-4071

www.lumenera.com

© 2009 Lumenera Corporation, all rights reserved.

Design, features, and specifications are subject to change without notice.